openEuler 24.03 LTS SP1 Technical White Paper

1 Introduction

The openEuler open source community is incubated and operated by the OpenAtom Foundation.

openEuler is a digital infrastructure OS that fits into any server, cloud computing, edge computing, and embedded deployment. This secure, stable, and easy-to-use open source OS is compatible with multiple computing architectures. openEuler suits operational technology (OT) applications and enables the convergence of OT and information and communications technology (ICT).

The openEuler open source community is a portal available to global developers, with the goal of building an open, diversified, and architecture-inclusive software ecosystem for all digital infrastructure scenarios. It has a rich history of helping enterprises develop their software, hardware, and applications.

The openEuler open source community was officially established on December 31, 2019, with the original focus of innovating diversified computing architectures.

On March 30, 2020, the Long Term Support (LTS) version openEuler 20.03 was officially released, which was a new Linux distribution with independent technology evolution.

Later in 2020, on September 30, the innovative openEuler 20.09 version was released through the collaboration efforts of multiple companies, teams, and independent developers in the openEuler community. The release of openEuler 20.09 marked a milestone not only in the growth of the openEuler community, but also in the history of open sourced software in China.

On March 31, 2021, the innovative kernel version openEuler 21.03 was released. This version is enhanced in line with Linux kernel 5.10 and also incorporates multiple new features, such as live kernel upgrade and tiered memory expansion. These features improve multi-core performance and deliver the computing power of one thousand cores.

Fast forward to September 30, 2021, openEuler 21.09 was released. This premium version is designed to supercharge all scenarios, including edge and embedded devices. It enhances server and cloud computing features, and incorporates key technologies including cloud-native CPU scheduling algorithms for hybrid service deployments and KubeOS for containers.

On March 30, 2022, openEuler 22.03 LTS was released based on Linux kernel 5.10. Designed to meet all server, cloud, edge computing, and embedded workloads, openEuler 22.03 LTS is an all-scenario digital infrastructure OS that unleashes premium computing power and resource utilization.

On September 30, 2022, openEuler 22.09 was released to further enhance all-scenario innovations.

On December 30, 2022, openEuler 22.03 LTS SP1 was released, which is designed for hitless porting with best-of-breed tools.

On March 30, 2023, openEuler 23.03 was released. Running on Linux kernel 6.1, it streamlines technical readiness for Linux kernel 6.x and facilitates innovations for hardware adaptation and other technologies.

On June 30, 2023, openEuler 22.03 LTS SP2 was released, which enhances scenario-specific features and increases system performance to a higher level.

Later on September 30, 2023, the openEuler 23.09 innovation version was released based on Linux kernel 6.4. It provides a series of brand-new features to further enhance developer and user experience.

On November 30, 2023, openEuler 20.03 LTS SP4 was released, an enhanced extension of openEuler 20.03 LTS. openEuler 20.03 LTS SP4 provides excellent support for server, cloud native, and edge computing scenarios.

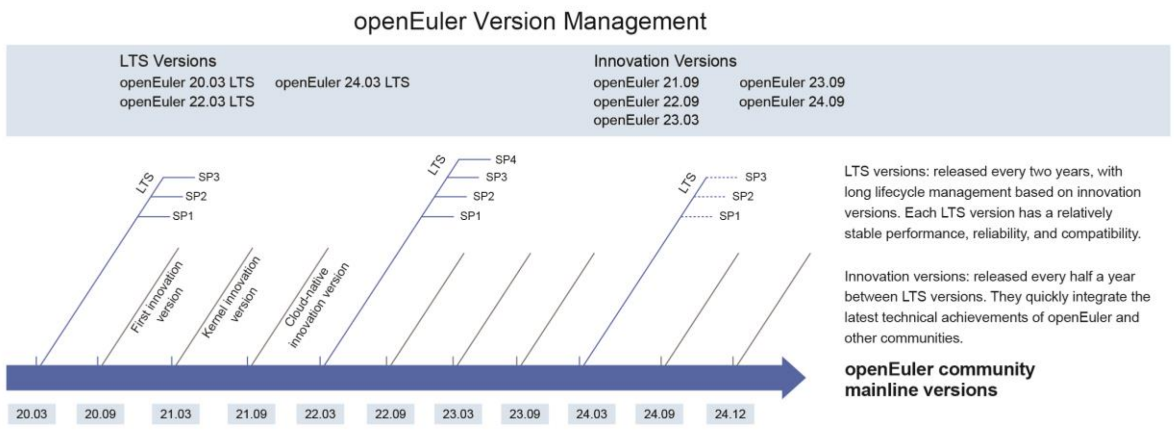
On December 30, 2023, openEuler 22.03 LTS SP3 was released. Designed to improve developer efficiency, it features for server, cloud native, edge computing, and embedded scenarios.

On May 30, 2024, openEuler 24.03 LTS was released. This version is built on Linux kernel 6.6 and brings new features for server, cloud, edge computing, AI, and embedded deployments to deliver enhanced developer and user experience.

On June 30, 2024, openEuler 22.03 LTS SP4 was released. Designed to improve developer efficiency, it further extends features for server, cloud native, edge computing, and embedded scenarios.

On September 30, 2024, openEuler 24.09 was released. It is an innovation version built based on Linux kernel 6.6 and brings more advanced features and functions.

More recently, openEuler 24.03 LTS SP1 was released on December 30, 2024. This enhanced and extended version of openEuler 24.03 LTS is developed on Linux kernel 6.6 and designed for server, cloud, edge computing, and embedded deployments. It offers new features and enhanced functions that streamline processes across a range of domains.



LTS versions: released every two years, with long lifecycle management based on innovation versions. Each LTS version has a relatively stable performance, reliability, and compatibility.

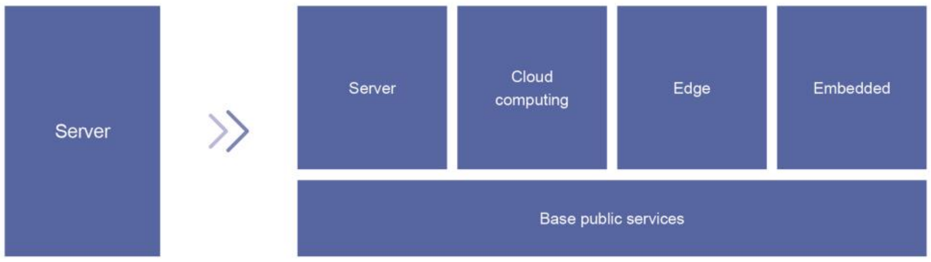
Innovation versions: released every half a year between LTS versions. They quickly integrate the latest technical achievements of openEuler and other communities.

As an OS platform, openEuler releases an LTS version every two years. Each LTS version provides enhanced specifications and a secure, stable, and reliable OS for enterprise users.

openEuler is built on tried-and-tested technologies. A new openEuler innovation version is released every 6 months to quickly integrate the latest technical achievements of openEuler and other communities. The innovative tech is first verified in the openEuler open source community as a single open source project, and then these features are added to each new release, enabling community developers to obtain the source code.

Technical capabilities are first tested in the open source community, and continuously incorporated into each openEuler release. In addition, each release is built on feedback given by community users to bridge the gap between innovation and the community, as well as improve existing technologies. openEuler is both a release platform and incubator of new technologies, working in a symbiotic relationship that drives the evolution of new versions.

Innovative Platform for All Scenarios



openEuler supports multiple processor architectures (x86, Arm, SW64, RISC-V, LoongArch, and PowerPC), as part of a focus to continuously improve the ecosystem of diversified computing power.

The openEuler community is home to an increasing number of special interest groups (SIGs), which are dedicated teams that help extend the OS features from server to cloud computing, edge computing, and embedded scenarios. openEuler is built to be used in any scenario, and comprises openEuler Edge and openEuler Embedded that are designed for edge computing and embedded deployments, respectively.

The OS is a perfect choice for ecosystem partners, users, and developers who plan to enhance scenario-specific capabilities. By creating a unified OS that supports multiple devices, openEuler hopes to enable a single application development for all scenarios.

Open and Transparent: Open Source Software Supply Chain

The process of building an open source OS relies on supply chain aggregation and optimization. To ensure reliable open source software or a large-scale commercial OS, openEuler comprises a complete lifecycle management that covers building, verification, and distribution. The brand regularly reviews its software dependencies based on user scenarios, organizes the upstream community addresses of all the software packages, and verifies its source code by comparing it to that of the upstream communities. The build, runtime dependencies, and upstream communities of the open source software form a closed loop, realizing a complete, transparent software supply chain management.

2 Platform Architecture

System Framework

openEuler is an innovative open source OS platform built on kernel innovations and a solid cloud base to cover all scenarios. It is built on the latest trends of interconnect buses and storage media, and offers a distributed, real-time acceleration engine and base services. It provides competitive advantages in edge and embedded scenarios, and is the first step to building an all-scenario digital infrastructure OS.

openEuler 24.03 LTS SP1 runs on Linux kernel 6.6 and provides POSIX-compliant APIs and OS releases for server, cloud native, edge, and embedded environments. It is a solid foundation for intelligent collaboration across hybrid and heterogeneous deployments. openEuler 24.03 LTS SP1 is equipped with a distributed soft bus and KubeEdge+ edge-cloud collaboration framework, among other premium features, making it a perfect choice for collaboration over digital infrastructure and everything connected models.

In the future, the openEuler open source community will continue to innovate, aiming to promote the ecosystem and consolidate the digital infrastructure.

Cloud base:

- **KubeOS for containers**: In cloud native scenarios, the OS is deployed and maintained in containers, allowing the OS to be managed based on Kubernetes, just as service containers.
- **Secure container solution**: Compared with the traditional Docker+QEMU solution, the iSulad+shimv2+StratoVirt secure container solution reduces the memory overhead and boot time by 40%.
- **Dual-plane deployment tool eggo**: OSs can be installed with one click for Arm and x86 hybrid clusters, while deployment of a 100-node cluster is possible within just 15 minutes.

New scenarios:

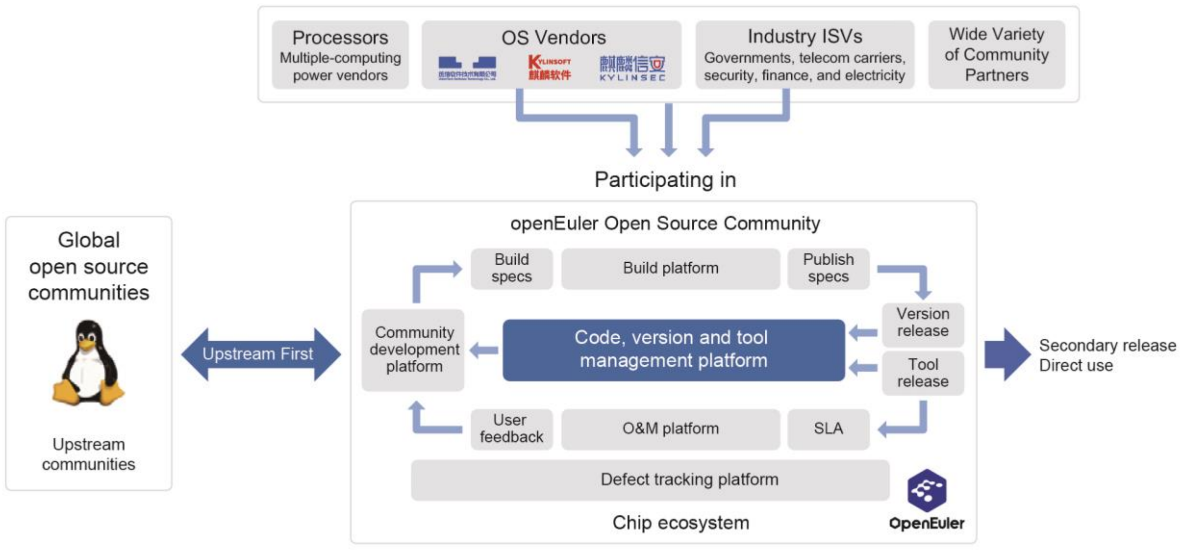
- **Edge computing**: openEuler 24.03 LTS SP1 Embedded is released for edge computing scenarios. It integrates the KubeEdge+ edge-cloud collaboration framework to provide unified management, provisioning of edge and cloud applications, and other capabilities.
- **Embedded**: openEuler 24.03 LTS SP1 Embedded is released for embedded scenarios, helping compress images to under 5 MB and shorten the image loading time to under 5 seconds.
- **AI-native OS**: The OS enables AI software stacks with out-of-the-box availability. Heterogeneous convergence of memory, scheduling, and training/inference resources reduces AI development costs and improves efficiency. The intelligent interaction platform of the OS streamlines development and administration.

Flourishing community ecosystem:

- **Desktop environments**: UKUI, DDE, Xfce, Kiran-desktop, and GNOME.
- **openEuler DevKit**: Supports OS migration, compatibility assessment, and various development tools such as secPaver which simplifies security configuration.

Platform Framework

The openEuler open source community partners with upstream and downstream communities to advance the evolution of openEuler versions.



Hardware Support

Visit https://www.openeuler.org/en/compatibility/ to see the full hardware list.

3 Operating Environments

Servers

To install openEuler on a physical machine, check that the physical machine meets the compatibility and hardware requirements.

For a full list, visit https://openeuler.org/en/compatibility/.

Item	Configuration Requirement
Architecture	AArch64, x86_64, RISC-V
Memory	≥ 4 GB
Drive	≥ 20 GB

VMs

Verify VM compatibility when installing openEuler.

Hosts running on openEuler 24.03 LTS SP1 support the following software packages:

- libvirt-9.10.0-12.oe2409
- libvirt-client-9.10.0-12.oe2409
- libvirt-daemon-9.10.0-12.oe2409
- qemu-8.2.0-17.oe2409
- qemu-img-8.2.0-17.oe2409

openEuler 24.03 LTS SP1 is compatible with the following guest OSs for VMs:

Guest OS	Architecture
CentOS 6	x86_64
CentOS 7	AArch64
CentOS 7	x86_64
CentOS 8	AArch64
CentOS 8	x86_64
Windows Server 2016	AArch64
Windows Server 2016	x86_64
Windows Server 2019	AArch64

Guest OS	Architecture
Windows Server 2019	x86_64

Item	Configuration Requirement
Architecture	AArch64, x86_64
CPU	≥ 2 CPUs
Memory	≥ 4 GB
Drive	≥ 20 GB

Edge Devices

To install openEuler on an edge device, check that the edge device meets the compatibility and minimum hardware requirements.

Item	Configuration Requirement
Architecture	AArch64, x86_64
Memory	≥ 4 GB
Drive	≥ 20 GB

Embedded Devices

To install openEuler Embedded on an embedded device, check that the embedded device meets the compatibility and minimum hardware requirements.

Item	Configuration Requirement
Architecture	AArch64, AArch32
Memory	≥ 512 MB
Drive	≥ 256 MB

4 Scenario-specific Innovations

AI

AI is redefining OSs by powering intelligent development, deployment, and O&M. openEuler supports general-purpose architectures like Arm, x86, and RISC-V, and next-gen AI processors like NVIDIA and Ascend. Further, openEuler is equipped with extensive AI capabilities that have made it a preferred choice for diversified computing power.

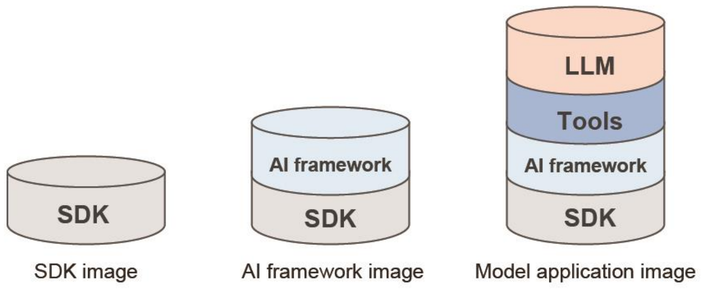
OS for AI

openEuler offers an efficient development and runtime environment that containerizes software stacks of AI platforms with out-of-the-box availability. It also provides various AI frameworks to facilitate AI development.

Feature Description

openEuler supports TensorFlow, PyTorch, and MindSpore frameworks and software development kits (SDKs) of major computing architectures, such as Compute Architecture for Neural Networks (CANN) and Compute Unified Architecture (CUDA), to make it easy to develop and run AI applications.

Environment setup is further simplified by containerizing software stacks. openEuler provides three types of container images:



- **SDK images**: Use openEuler as the base image and install the SDK of a computing architecture, for example, Ascend CANN and NVIDIA CUDA.
- **AI framework images**: Use an SDK image as the base and install AI framework software, such as PyTorch and TensorFlow. You can use an AI framework image to quickly build a distributed AI framework, such as Ray.
- **Model application images**: Provide a complete set of toolchains and model applications.

For details, see the openEuler AI Container Image User Guide.

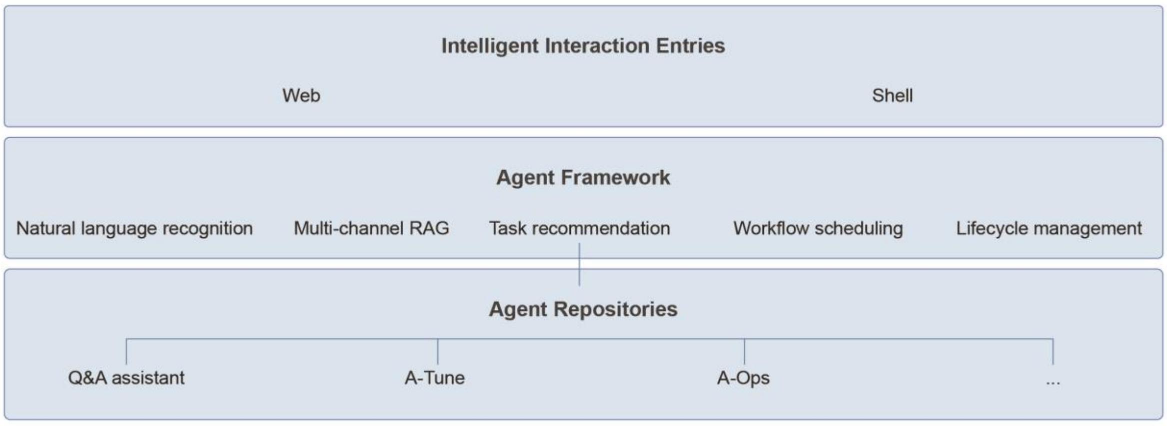
Application Scenarios

openEuler uses AI container images to simplify deployment of runtime environments. You can select the container image that best suits your requirements and complete the deployment in a few simple steps.

- **SDK images**: You can develop and debug Ascend CANN or NVIDIA CUDA applications using an SDK image, which provides a compute acceleration toolkit and a development environment. These containers offer an easy way to perform high-performance computing (HPC) tasks, such as large-scale data processing and parallel computing.
- **AI framework images**: This type of containers is designed to support AI model development, training, and inference.
- **Model application images**: Such an image contains a complete AI software stack and purpose-built models for model inference and fine-tuning.

AI for OS

AI is making openEuler smarter. The openEuler Copilot System is an intelligent Q&A platform developed on foundation models and openEuler data. It is designed to streamline code generation, troubleshooting, and O&M.



Intelligent Q&A

Feature Description

The openEuler Copilot System is accessible via web or shell.

- **Web**: The easy-to-use chatbot provides access to openEuler knowledge, community announcements, and O&M solutions. It uses various intelligent agents to deliver user-friendly functionality.
- **Shell**: This method allows users to interact with openEuler using human language, providing heuristic system tuning and fault diagnosis for easier system management.

Workflow Scheduling

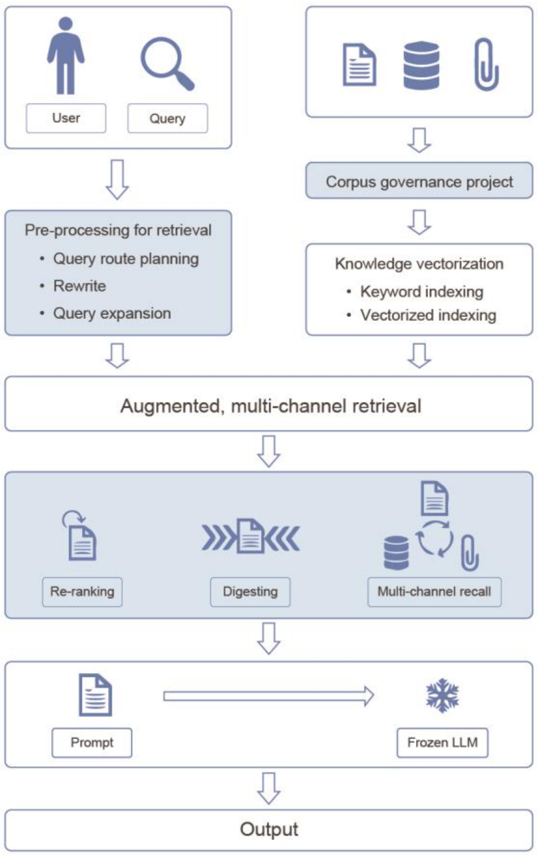
- **Atomic agent operations**: Multiple agent operations can be combined into a multi-step workflow that is internally ordered and associated, and is executed as an inseparable atomic operation. By supporting user-defined error recovery, this design ensures consistent, traceable, and stable operations when a complex task is executed.
- **Real-time data processing**: Data generated in each step of the workflow can be processed immediately and then transferred to the next step. Users can transfer information and access required data between steps with ease. The final results can be displayed on the front end in multiple forms, such as reports, images, and code snippets.
- **Intelligent interaction**: When the openEuler Copilot System receives a vague or complex user instruction, it proactively asks the user to clarify and provide more details. With new information, the system continues to execute the workflow and meet the user's expectations.

Task Recommendation

- **Intelligent response**: The openEuler Copilot System can analyze the semantic information entered in text format, determining the user's intent and selecting the most matched workflow. The system learns and adapts to users' operation habits and preferences to improve user experience.

- **Intelligent guidance**: The openEuler Copilot System comprehensively analyzes the execution status, function requirements, and associated tasks of the current workflow, and provides next-step operation suggestions based on users' personal preferences and historical patterns. This helps inform user decisions and improves task efficiency on an intuitive front-end display.

RAG



Retrieval-augmented generation (RAG) is a technique for enhancing the long-term memory capability of large language models (LLMs). It is used in the openEuler Copilot System to reduce model training costs. The RAG technique has the following highlights:

- **Pre-processing for retrieval**: Users' compound query requests are rewritten and routed based on a preset plan.
- **Knowledge indexing**: Keywords are extracted and text is vectorized for diversified document content, and indexes can be built for fragments.
- **Multi-channel recall**: Vectorized retrieval, keyword retrieval, and Chat2DB capabilities are provided for diversified knowledge sources. Retrieval results can be re-ranked, filtered, and converged.

These capabilities enable the openEuler Copilot System to support more document formats and content scenarios, and enhance Q&A quality while adding little system load.

Corpus Governance

Corpus governance is a core RAG capability. It imports corpuses into the knowledge base in a supported format using fragment relationship extraction, fragment derivative construction, and optical character recognition (OCR). This increases the retrieval hit rate.

- **Fragment relationship extraction**: In scenarios where document content continuity must be maintained, relative relationships between fragments are retained to associate context.
- **Fragment derivative generation**: Digests are provided for complicated fragments, making it possible to hit fragments through these digests.
- **OCR**: OCR and contextual digests are available for hybrid image and text scenarios, which allow extracting and abstracting text from images.

Corpus governance features enhance the Q&A experience in multi-round dialogs, content integrity, and image-text display.

Application Scenarios

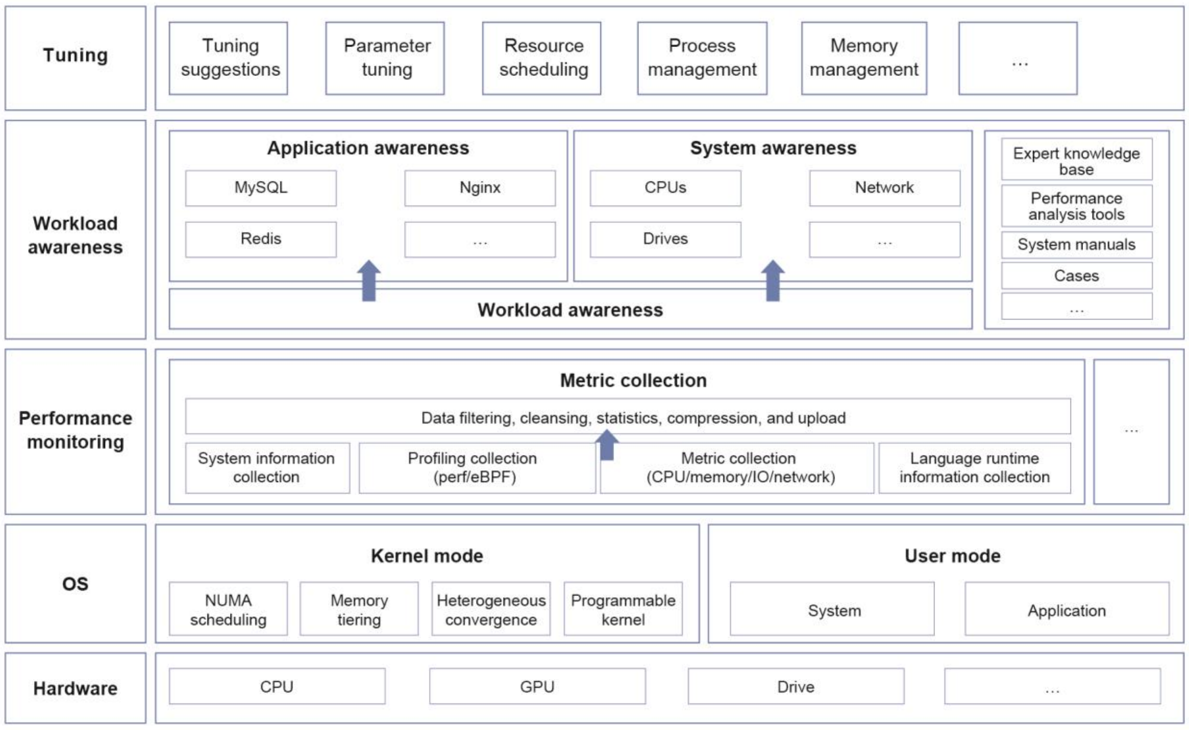
- **Common users** who are seeking best practices, such as porting applications to openEuler.
- **Developers** who want to learn contribution processes, key features, project development, and other extensive knowledge of openEuler.
- **O&M personnel** who aim to solve common problems and improve system management based on Q&A system suggestions.

For details, see the openEuler Copilot System Intelligent Q&A Service User Guide.

Intelligent Tuning

Feature Description

The openEuler Copilot System supports the intelligent shell entry. Through this entry, you can interact with the openEuler Copilot System using a natural language and perform heuristic tuning operations such as performance data collection, system performance analysis, and system performance tuning.

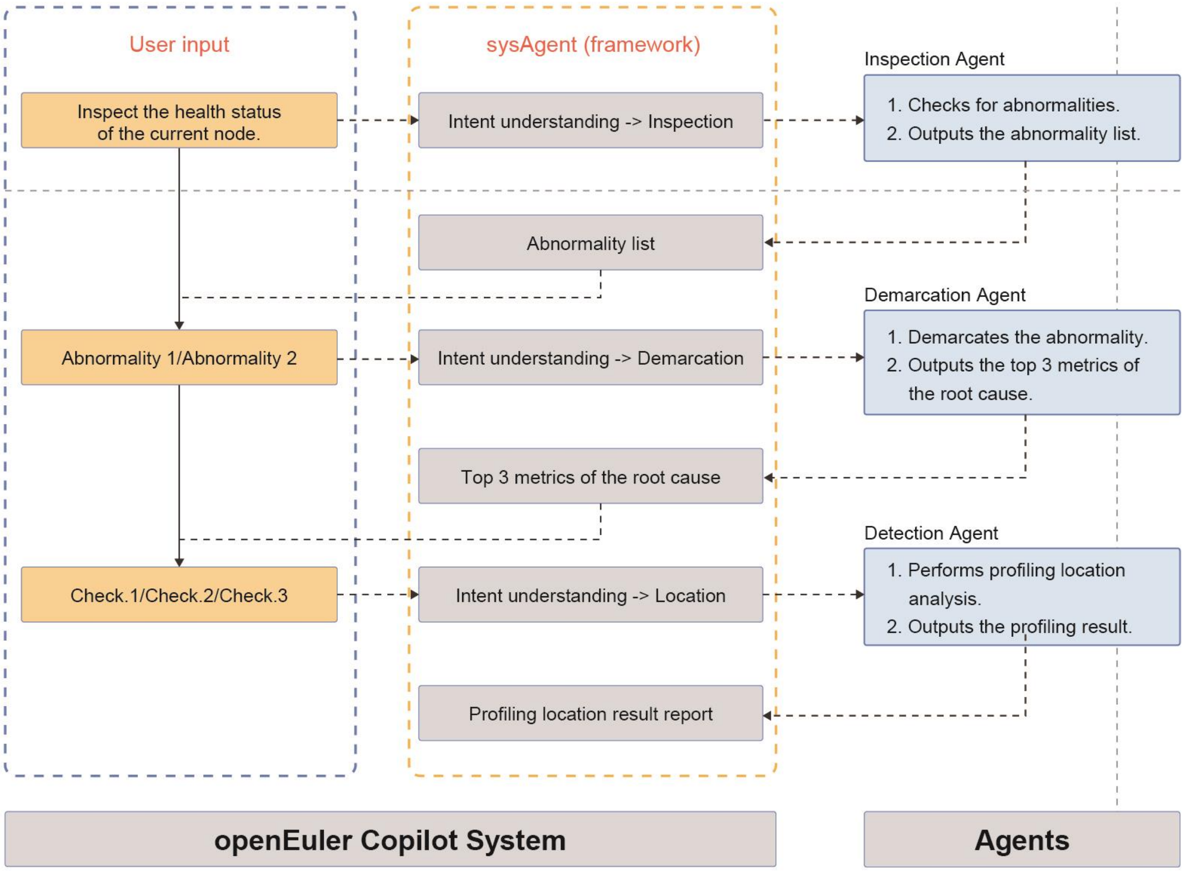


Application Scenarios

- **Gaining insights from key performance metrics**: You can learn about the system performance status based on collected performance metrics like CPU, I/O, drive, network, and application.
- **Analyzing system performance**: Performance analysis reports are generated, making it easier to locate performance bottlenecks across the entire system and in individual applications.
- **Receiving performance tuning suggestions**: The openEuler Copilot System generates a performance tuning script, which can be executed with one click to tune the system and specific applications.

Intelligent Diagnosis

Feature Description


User input
sysAgent (framework)
Inspection Agent
Inspect the health status of the current node.
Intent understanding -> Inspection
1. Checks for abnormalities.
2. Outputs the abnormality list.
Abnormality list
Demarcation Agent
Abnormality 1/Abnormality 2
Intent understanding -> Demarcation
1. Demarcates the abnormality.
2. Outputs the top 3 metrics of the root cause.
Top 3 metrics of the root cause
Detection Agent
Check.1/Check.2/Check.3
Intent understanding -> Location
1. Performs profiling location analysis.
2. Outputs the profiling result.
Profiling location result report
openEuler Copilot System
Agents

- **Inspection**: The Inspection Agent checks for abnormalities of designated IP addresses and provides an abnormality list that contains associated container IDs and abnormal metrics (such as CPU and memory).
- **Demarcation**: The Demarcation Agent analyzes and demarcates a specified abnormality contained in the inspection result and outputs the top 3 metrics of the root cause.
- **Location**: The Detection Agent performs profiling location analysis on the root cause, and provides useful hotspot information such as the stack, system time, and performance metrics related to the root cause.

Application Scenarios

In openEuler 24.03 LTS SP1, the intelligent shell entry enables capabilities like single-node abnormality inspection, demarcation, and profiling location.

- The inspection capabilities refer to single-node performance metric collection, performance analysis, and abnormality inspection.
- The demarcation capability is to locate the root cause based on the abnormality inspection result and output the top 3 metrics of the root cause.

- The profiling location capability refers to using a profiling tool to locate the faulty modules (code snippets) based on the root cause.

Intelligent Container Images

Feature Description

The openEuler Copilot System can invoke environment resources through a natural language, assist in pulling container images for local physical resources, and establish a development environment suitable for debugging on existing compute devices.

This system supports three types of containers, and container images have been released on Docker Hub. You can manually pull and run these container images.

- **SDK layer**: encapsulates only the component libraries that enable AI hardware resources, such as CUDA and CANN.
- **SDKs + training/inference frameworks**: accommodates TensorFlow, PyTorch, and other frameworks (for example, tensorflow2.15.0-cuda12.2.0 and pytorch2.1.0.a1-cann7.0.RC1) in addition to the SDK layer.
- **SDKs + training/inference frameworks + LLMs**: encapsulates several models (for example, llama2-7b and chatglm2-13b) based on the second type of containers.

The following table lists the container images supported by the openEuler Copilot System:

Registry	Repository	Image Name	Tag
docker.io	openeuler	cann	8.0.RC1-oe2203sp4
			cann7.0.RC1.alpha002-oe2203sp2
docker.io	openeuler	oneapi-runtime	2024.2.0-oe2403lts
docker.io	openeuler	oneapi-basekit	2024.2.0-oe2403lts
docker.io	openeuler	llm-server	1.0.0-oe2203sp3
docker.io	openeuler	mlflow	2.11.1-oe2203sp3
			2.13.1-oe2203sp3
docker.io	openeuler	llm	chatglm2_6b-pytorch2.1.0.a1-cann7.0.RC1.alpha002-oe2203sp2
			llama2-7b-q8_0-oe2203sp2
			chatglm2-6b-q8_0-oe2203sp2
			fastchat-pytorch2.1.0.a1-cann7.0.RC1.alpha002-oe2203sp2
docker.io	openeuler	tensorflow	tensorflow2.15.0-oe2203sp2
			tensorflow2.15.0-cuda12.2.0-devel-cudnn8.9.5.30-oe2203sp2
docker.io	openeuler	pytorch	pytorch2.1.0-oe2203sp2

Registry	Repository	Image Name	Tag
			pytorch2.1.0-cuda12.2.0-devel-cudnn8.9.5.30-oe2203sp2
			pytorch2.1.0.a1-cann7.0.RC1.alpha002-oe2203sp2
docker.io	openeuler	cuda	cuda12.2.0-devel-cudnn8.9.5.30-oe2203sp2

Application Scenarios

- **Common openEuler operations**: Simplify the process of building a deep learning development environment while saving physical resources. For example, set up an Ascend development environment on openEuler.
- **openEuler development**: Developers familiarize themselves with the openEuler AI software stack to reduce the trial-and-error cost of installing components.

openEuler Embedded

openEuler 24.03 LTS SP1 Embedded is designed for embedded applications, offering significant progress in southbound and northbound ecosystems, technical features, infrastructure, and implementation over previous generations.

openEuler Embedded provides a closed loop framework often found in operational technology (OT) applications such as manufacturing and robotics, whereby innovations help optimize its embedded system software stack and ecosystem.

openEuler Embedded supports Linux kernels 5.10 and 6.6, and users can select either Linux kernel or customize a kernel to better fulfill their board support package (BSP) and application requirements.

openEuler Embedded is equipped with an embedded virtualization base that is available as multiple solutions, including Jailhouse partitioning-based virtualization, OpenAMP bare metal hybrid deployment, and Zephyr-based Virtual Machine (ZVM) & Rust-Shyper real-time virtualization. You can select the most appropriate solution to suit your services. The mixed-criticality (MICA) deployment framework is built on the embedded virtualization base to mask discrepancies between different bases and provides a unified set of APIs for different runtimes.

The MICA deployment framework supports over 600 northbound software packages, including the ROS Humble version. ROS Humble contains ros-core, ros-base, and SLAM software packages to implement the ROS 2 runtime. SDKs with embedded features of ROS 2 are provided for ROS 2 colcon cross-compilation. openEuler Embedded is already included in the BMC ecosystem and has been adapted to openBMC.

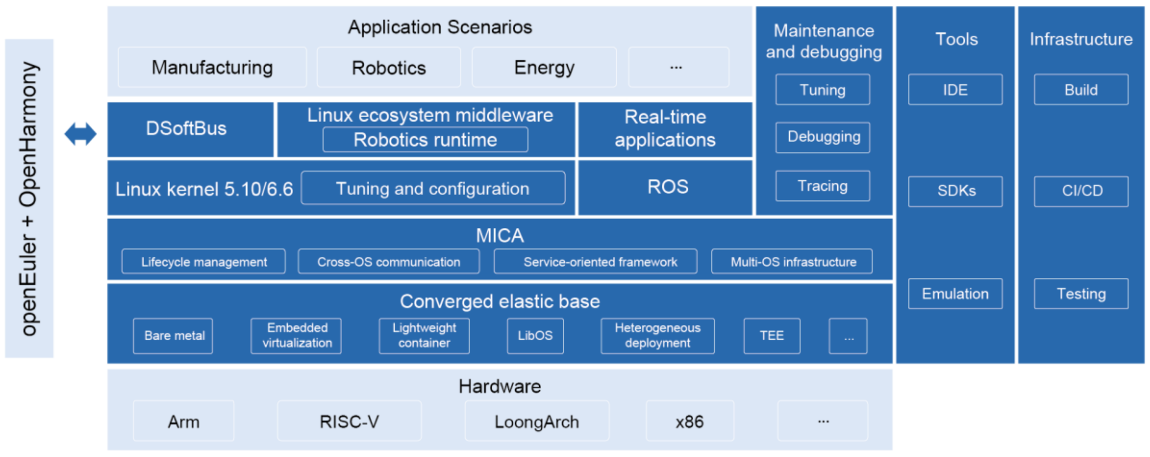
The MICA deployment framework supports a variety of southbound hardware, including Phytium, HiSilicon, Renesas, TI, and Allwinner, and also EulerPi (hardware development board designed for developers) and HiEuler Pi (native development board for openEuler Embedded).

Infrastructure tools available on openEuler Embedded include the meta-tool oebuild and build tool Yocto 4.0 LTS. In addition, the LLVM toolchain is introduced to help better build images and generate SDKs with its advantages in performance, size, and security.

openEuler Embedded has multiple commercial and enterprise editions, which have been applied in BMC, industrial controller, robot controller, and other fields.

Future versions of openEuler Embedded will integrate contributions from ecosystem partners, users, and community developers, increase support for chip architectures such as LoongArch and more southbound hardware, and optimize industrial middleware, embedded AI, embedded edge, and simulation system capabilities.

System Architecture



Southbound Ecosystem

openEuler Embedded Linux supports mainstream processor architectures like AArch64, x86_64, AArch32, and RISC-V, and will extend support to LoongArch in the future. openEuler 24.03 and later versions have a rich southbound ecosystem and support chips from Raspberry Pi, HiSilicon, Rockchip, Renesas, TI, Phytium, StarFive, and Allwinner. In openEuler 24.03 LTS SP1 Embedded, Kunpeng 920 is also supported.

Embedded Virtualization Base

openEuler Embedded uses an elastic virtualization base that enables multiple OSs to run on a system-on-a-chip (SoC). The base incorporates a series of technologies including bare metal, embedded virtualization, lightweight containers, LibOS, trusted execution environment (TEE), and heterogeneous deployment.

- The bare metal hybrid deployment solution runs on OpenAMP to manage peripherals by partition at a high performance level; however, it delivers poor isolation and flexibility. This solution supports the hybrid deployment of UniProton/Zephyr/RT-Thread and openEuler Embedded Linux.
- Partitioning-based virtualization is an industrial-grade hardware partition virtualization solution that runs on Jailhouse. It offers superior performance and isolation but inferior flexibility. This solution supports the hybrid deployment of UniProton/Zephyr/FreeRTOS and openEuler Embedded Linux or of OpenHarmony and openEuler Embedded Linux.
- Real-time virtualization is available as two community hypervisors, ZVM (for real-time VM monitoring) and Rust-Shyper (for Type-I embedded VM monitoring).

MICA Deployment Framework

The MICA deployment framework is a unified environment that masks the differences between technologies that comprise the embedded elastic virtualization base. The multi-core capability of hardware combines the universal Linux OS and a dedicated real-time operating system (RTOS) to make full use of all OSs.

The MICA deployment framework covers lifecycle management, cross-OS communication, service-oriented framework, and multi-OS infrastructure.

- Lifecycle management provides operations to load, start, suspend, and stop the client OS.
- Cross-OS communication uses a set of communication mechanisms between different OSs based on shared memory.
- Service-oriented framework enables different OSs to provide their own services. For example, Linux provides common file system and network services, and the RTOS provides real-time control and computing.
- Multi-OS infrastructure integrates OSs through a series of mechanisms, covering resource expression and allocation and unified build.

The MICA deployment framework provides the following functions:

- Lifecycle management and cross-OS communication for openEuler Embedded Linux and the RTOS (Zephyr or UniProton) in bare metal mode
- Lifecycle management and cross-OS communication for openEuler Embedded Linux and the RTOS (FreeRTOS or Zephyr) in partitioning-based virtualization mode

Northbound Ecosystem

- **Northbound software packages**: Over 600 common embedded software packages can be built using openEuler.
- **Soft real-time kernel**: This capability helps respond to soft real-time interrupts within microseconds.
- **DSoftBus**: The distributed soft bus system (DSoftBus) of openEuler Embedded integrates the DSoftBus and point-to-point authentication module of OpenHarmony. It implements interconnection between openEuler-based embedded devices and OpenHarmony-based devices as well as between openEuler-based embedded devices.
- **Embedded containers and edges**: With iSula containers, openEuler and other OS containers can be deployed on embedded devices to simplify application porting and deployment. Embedded container images can be compressed to 5 MB, and can be easily deployed into the OS on another container.

UniProton

UniProton is an RTOS that features ultra-low latency and flexible MICA deployments. It is suited for industrial control because it supports both microcontroller units and multi-core CPUs. UniProton provides the following capabilities:

- Compatible with processor architectures like Cortex-M, AArch64, x86_64, and riscv64, and supports M4, RK3568, RK3588, x86_64, Hi3093, Raspberry Pi 4B, Kunpeng 920, Ascend 310, and Allwinner D1s.
- Connects with openEuler Embedded Linux on Raspberry Pi 4B, Hi3093, RK3588, and x86_64 devices in bare metal mode.
- Can be debugged using GDB on openEuler Embedded Linux.

Application Scenarios

openEuler Embedded helps supercharge computing performance in a wide range of industries and fields, including industrial and power control, robotics, aerospace, automobiles, and healthcare.

DevStation

DevStation is the first openEuler developer workstation to come pre-installed with VS Code. Designed to streamline software coding, compilation, build, release, and deployment, DevStation integrates the oeDeploy tool to simplify the deployment of the AI and cloud-native software stacks and uses oeDevPlugin to pull code repositories and the AI4C-powered compiler for compilation. Later versions will include epkg, a new openEuler package manager that enables developers to seamlessly manage and switch between multiple software versions across different environments.

Feature Description

Developer-friendly integrated environment: supports multiple programming languages and comes with pre-installed development tools and IDEs such as VS Code to streamline front-end, back-end, and full-stack development.

GUI-based programming: realizes intuitive programming.

AI development: integrates oeDeploy to install Kubeflow, reducing costs.

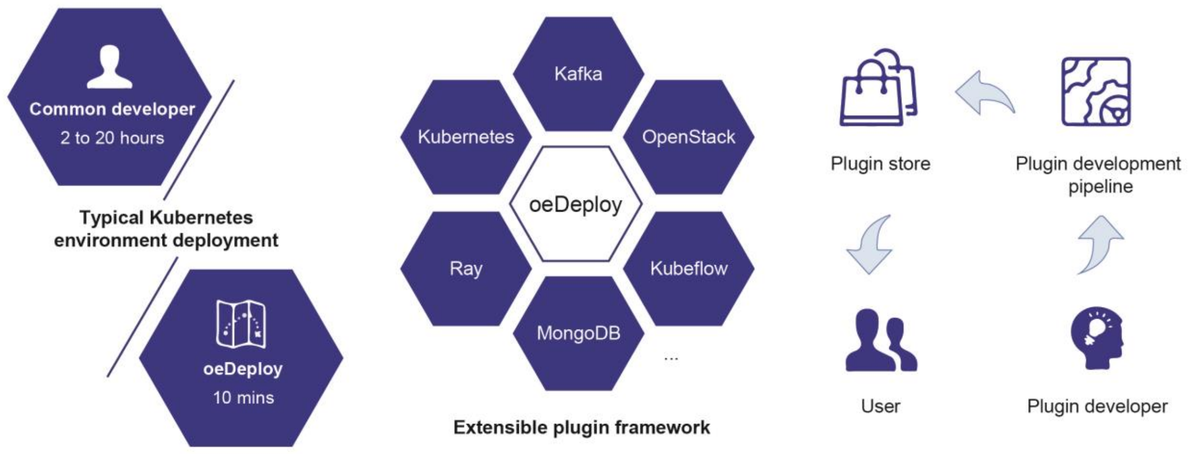
Debugging and testing: provides built-in debugging tools, such as GDB, CUnit, GTest, and perf, and test and tuning tools for fast debugging and automated testing

Versioning and collaboration: integrates the Git versioning tool and remote collaboration tools such as Slack, Mattermost, and GitLab, promoting team development and remote collaboration.

Application Scenarios

Multi-language development: DevStation is suitable for projects employing multiple languages such as Python, JavaScript, Java, and C++. With various pre-installed compilers, interpreters, and build tools, developers do not need to manually configure the environment.

Quick installation and deployment: DevStation integrates oeDeploy to deploy distributed software such as Kubeflow and Kubernetes within minutes. oeDeploy provides a unified plugin framework and atomic deployment capabilities to streamline installation and deployment. Developers can quickly release custom installation and deployment plugins by following simple development specifications.



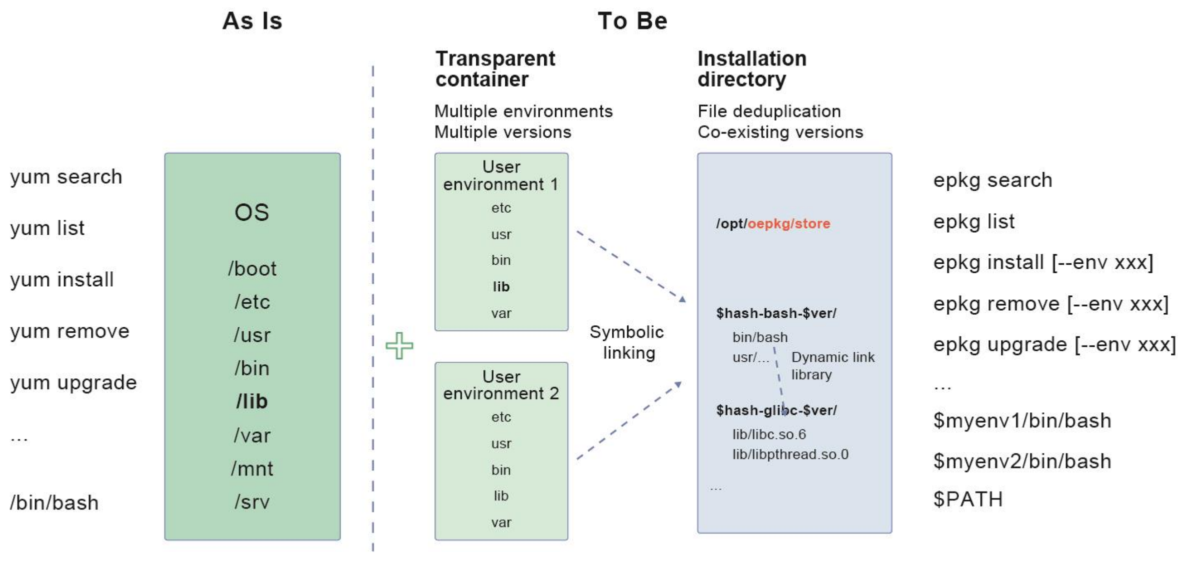
AI development and data science: Pre-installed databases, model training tools, and heterogeneous computing accelerators combine to create a complete AI development environment for model training and inference.

Game and multimedia development: DevStation comes with multimedia editing software to support image, audio, and video processing.

Programming training: DevStation is suitable for both entry-level and senior developers, as it allows GUI-based programming, provides built-in development tutorials, and will offer intelligent Q&A services about Linux distributions using the openEuler Copilot System.

epkg

epkg is a new software package manager that supports the installation and use of non-service software packages. It solves version compatibility issues so that users can install and run software of different versions on the same OS by using simple commands to create, enable, and switch between environments.



Feature Description

Version compatibility: enables multiple versions of the same software package to run on the same node without version conflicts.

Environment management: allows users to create, enable, and switch between environments. Users can use channels of different environments to use software packages of different versions. When an environment is switched to another, the software package version is also changed.

Installation by common users: allows common users to install software packages, create environments, and manage their environment images, reducing security risks associated with software package installation.

Application Scenarios

epkg solves compatibility issues in installing multiple versions of the same software package. Users can switch between environments to use different package versions.

For details, refer to the epkg User Guide.

openEuler GCC Toolset 14

openEuler GCC Toolset 14 offers more flexible and efficient compilation environments. Users can easily switch between different GCC versions to access new hardware features and improved performance brought by latest GCC optimizations.

Feature Description

To enable new compute features and make the most of hardware features, openEuler GCC Toolset 14 offers a minor GCC 14 toolchain that is later than the major GCC of openEuler 24.03 LTS SP1, enabling users to select the most appropriate compilation environment. By using openEuler GCC Toolset 14, users can easily switch between different GCC versions to use new hardware features.

To decouple from the default major GCC and prevent conflicts between the dependency libraries of the minor and major GCC versions, the software package of the openEuler GCC Toolset 14 toolchain is named starting with **gcc-toolset-14-**, followed by the name of the original GCC software package.

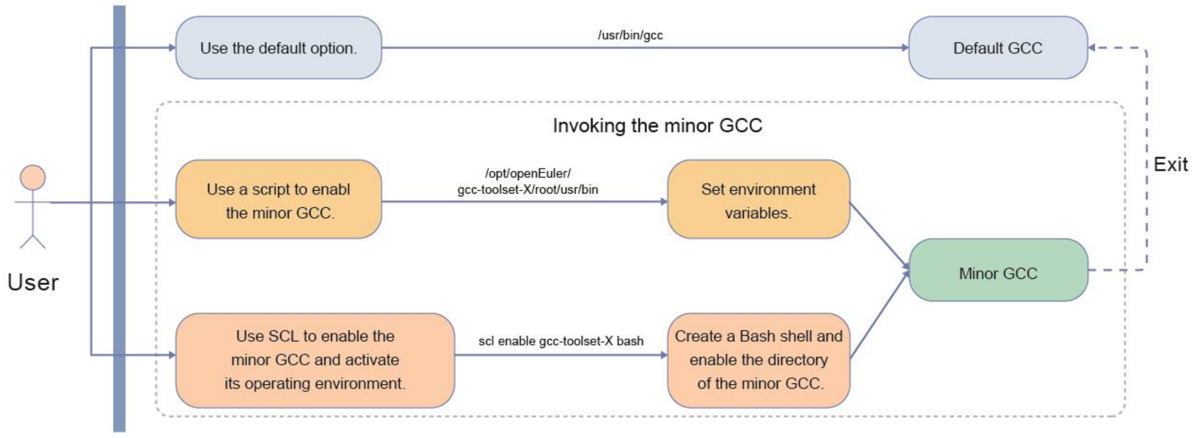
For version management, this solution introduces the SCL tool that provides an **enable** script in the **/opt/openEuler/gcc-toolset-14** directory to register the environment variables of openEuler GCC Toolset 14 with SCL. Then, users can use the tool to start a new Bash shell that uses the environment variables of the minor version configured in the **enable** script. In this way, it is easy to switch between the major and minor GCC versions.

Application Scenarios

Major GCC: The default GCC 12.3.1 is used for compilation.

Minor GCC: If the features of GCC 14 are required to build applications, use SCL to switch to the build environment of openEuler GCC Toolset 14.

The following figure shows the method of switching between the major and minor GCC:


User
Use the default option.
/usr/bin/gcc
Default GCC
Invoking the minor GCC
Use a script to enabl the minor GCC.
/opt/openEuler/ gcc-toolset-X/root/usr/bin
Set environment variables.
Minor GCC
Exit
Use SCL to enable the minor GCC and activate its operating environment.
scl enable gcc-toolset-X bash
Create a Bash shell and enable the directory of the minor GCC.

5 Kernel Innovations

New Features in the openEuler Kernel

openEuler 24.03 LTS SP1 runs on Linux kernel 6.6 and inherits the competitive advantages of community versions and innovative features released in the openEuler community.

- **Folio-based memory management**: Folio-based Linux memory management is used instead of page. A folio consists of one or more pages and is declared in **struct folio**. Folio-based memory management is performed on one or more complete pages, rather than on *PAGE_SIZE* bytes. This alleviates compound page conversion and tail page misoperations, while decreasing the number of least recently used (LRU) linked lists and optimizing memory reclamation. It allocates more continuous memory on a per-operation basis to reduce the number of page faults and mitigate memory fragmentation. Folio-based management accelerates large I/Os and improves throughput, and large folios consisting of anonymous pages or file pages are available. For AArch64 systems, a contiguous bit (16 contiguous page table entries are cached in a single entry within a translation lookaside buffer, or TLB) is provided to reduce system TLB misses and improve system performance. In openEuler 24.03 LTS SP1, multi-size transparent hugepage (mTHP) allocation by anonymous shmem and mTHP lazyfreeing are available. The memory subsystem supports large folios, with a new sysfs control interface for allocating mTHPs by page cache and a system-level switch for feature toggling.
- **Multipath TCP (MPTCP)**: MPTCP is introduced to let applications use multiple network paths for parallel data transmission, compared with single-path transmission over TCP. This design improves network hardware resource utilization and intelligently allocates traffic to different transmission paths, thereby relieving network congestion and improving throughput.

  MPTCP features the following performance highlights:

  - Selects the optimal path after evaluating indicators such as latency and bandwidth.
  - Ensures hitless network switchover and uninterrupted data transmission when switching between networks.
  - Uses multiple channels where data packets are distributed to implement parallel transmission, increasing network bandwidth.

  In the lab environment, the Rsync file transfer tool that adopts MPTCP v1 shows good transmission efficiency improvement. Specifically, a 1.3 GB file can be transferred in just 14.35s (down from 114.83s), and the average transfer speed is increased from 11.08 MB/s to 88.25 MB/s. In simulations of path failure caused by unexpected faults during transmission, MPTCP seamlessly switches data to other available channels, ensuring transmission continuity and data integrity.

In openEuler 24.03 LTS SP1, MPTCP-related features in Linux mainline kernel 6.9 have been fully transplanted and optimized.

- **Large folio for ext4 file systems**: The IOzone performance can be improved by 80%, and the writeback process of the iomap framework supports batch block mapping. Blocks can be requested in batches in default ext4, optimizing ext4 performance in various benchmarks. For ext4 buffer I/O and page cache writeback operations, the buffer_head framework is replaced with the iomap framework that adds large folio support for ext4. In version 24.03 LTS SP1, the performance of small buffered I/Os (≤ 4 KB) is optimized when the block size is smaller than the folio size, typically seeing a 20% performance increase.
- **xcall and xint**: The Linux kernel is becoming increasingly complex, and system calls, especially the simple ones, can be a performance bottleneck. System calls on the AArch64 platform share the same exception entry point, which includes redundant processes such as security checks. Common ways to reduce system call overhead include service offloading and batch processing, but both require service adaptation. xcall provides a solution that does not require service code modification. It streamlines system calls by optimizing their processing logic, trading off some maintenance and security capabilities to reduce overhead.

  To unify interrupt handling, the kernel integrates all interrupt handling processes into its general interrupt handling framework. As the kernel evolves, the general interrupt handling framework has gradually accumulated many security hardening and maintenance features that are not closely related to interrupt handling, increasing latency unpredictability. xint simplifies interrupt handling to reduce the latency and system overhead.
- **CacheFiles failover**. In on-demand mode of CacheFiles, if the daemon breaks down or is killed, subsequent read and mount requests return **-EIO**. The mount points can be used only after the daemon is restarted and the mount operations are performed again. For public cloud services, such I/O errors will be passed to cloud service users, which may impact job execution and endanger the overall system stability. The CacheFiles failover feature renders it unnecessary to remount the mount points upon daemon crashes. It requires only the daemon to restart, ensuring that these events are invisible to users.
- **Programmable scheduling**: Interfaces of programmable scheduling are available in the user space as an extension of the Completely Fair Scheduler (CFS) algorithm. Users can customize scheduling policies based on service scenarios to bypass the original CFS. This new feature allows programmable core selection, load balancing, task selection, and task preemption, and supports labelling and kfuncs.
- **SMC-D with loopback-ism**: Shared memory communication over DMA (SMC-D) is a kernel network protocol stack that is compatible with socket interfaces and accelerates TCP communication transparently over shared memory. Initially, SMC-D was exclusive to IBM S/390 architecture systems. The introduction of the loopback-ism virtual device enabled broader adoption by simulating Internal Shared Memory (ISM). This innovation transformed SMC-D into a universal kernel mechanism compatible with non-S/390 architectures as well. SMC-D with loopback-ism applies to scenarios where TCP is used for inter-process or inter-container communication in the OS. It accelerates communication by bypassing the kernel TCP/IP stack. Together with smc-tools, users can preload dynamic libraries through **LD_PRELOAD** to replace the TCP stack without adapting applications. Feedback from the community shows that, compared with the native TCP, SMC-D with loopback-ism can improve the network throughput by more than 40%.
- **IMA RoT framework**: The Linux Integrity Measurement Architecture (IMA) subsystem mainly uses the trusted platform module (TPM) as the root of trust (RoT) device to provide integrity proof for the measurement list, and code of the subsystem is tightly coupled with TPM operations. However, workloads like confidential computing require IMA to use new RoT devices, such as virtCCA supported by openEuler. This IMA RoT framework implements an

abstraction layer between the IMA subsystem and RoT devices. It simplifies the adaptation of RoT devices to the IMA subsystem and facilitates the configuration and operation of various RoT devices by users and the IMA subsystem.

- **Protection against script viruses**: Typical ransomware variants are script files (such as JSP files), which can run through the interpreter and bypass security technologies in the kernel to launch attacks. However, the IMA technology used by the kernel to defend against intrusion mainly targets virus files of the Executable and Linkable Format (ELF). This new feature enables IMA to check script files that are indirectly executed in the system. It uses the **execveat()** system call with newly added flags to check the execution permission of scripts, and call the IMA interface during the check to measure script integrity. Test and verification results demonstrate how the script interpreter proactively invokes **execveat()** and transfers the **AT_CHECK** flag to check the execute permission (including the IMA check) of scripts, which are executed only after they pass the permission check.
- **Halt polling**: This feature enables guest vCPUs to poll at the idle state to avoid sending an inter-processor interrupt (IPI) when performing a wakeup. This optimization reduces the interrupt sending and handling overhead. In addition, the VM does not exit during polling, further reducing the VM entry/exit overhead. This feature also reduces the inter-process communication latency and improves VM performance.

CAQM for the Kernel TCP/IP Stack

Constrained active queue management (CAQM) is an algorithm for network congestion control. It is mainly used on compute nodes in data centers that use TCP to transmit data and network switch nodes on transmission paths.

The network switch nodes calculate idle bandwidth and optimal bandwidth allocation, and the compute node protocol stack works with the switches to achieve "zero-queue" congestion control effect and ultra-low transmission latency in high-concurrency scenarios.

The CAQM algorithm adds congestion control flags to the Ethernet link layer to dynamically adjust the queue length and improve the utilization of network resources. In latency-sensitive general-computing scenarios in data centers, this feature greatly reduces latency and packet loss, improving user experience.

In typical data center scenarios, this algorithm outperforms the classic Cubic algorithm in two key metrics: (1) 92.4% lower transmission latency, and (2) 99.97% TCP transmission bandwidth utilization with 90% lower switch queue buffer usage.

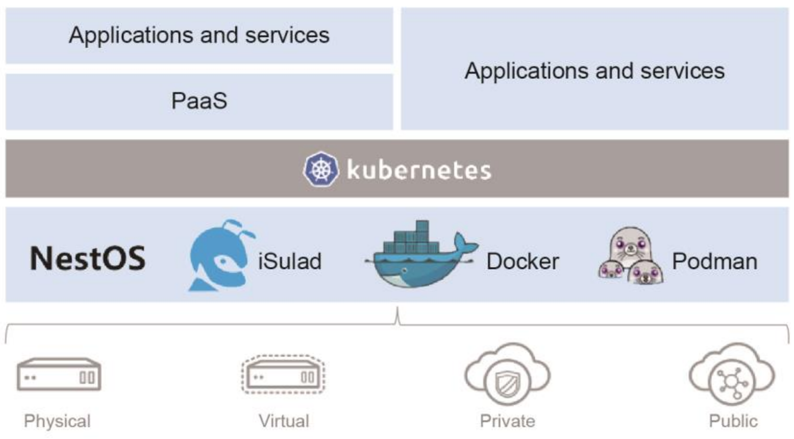
The CAQM algorithm requires collaboration between compute node servers and switches. Therefore, intermediate switches must support the CAQM protocol (for protocol header identification, congestion control field adjustment, and so on). This algorithm is controlled through the kernel compilation macro (**CONFIG_ETH_CAQM**) and is disabled by default. To use this algorithm, users need to enable this macro and recompile the kernel.

6 Cloud Base

NestOS

NestOS is a community cloud OS that uses nestos-assembler for quick integration and build. It runs rpm-ostree and Ignition tools over a dual rootfs and atomic update design, and enables easy cluster setup in large-scale containerized environments. Compatible with Kubernetes and OpenStack, NestOS also reduces container overheads.

Feature Description



- **Out-of-the-box availability**: integrates popular container engines such as iSulad, Docker, and Podman to provide lightweight and tailored OSs for the cloud.
- **Easy configuration**: uses the Ignition utility to install and configure a large number of cluster nodes with a single configuration.
- **Secure management**: runs rpm-ostree to manage software packages and works with the openEuler software package source to ensure secure and stable atomic updates.
- **Hitless node updating**: uses Zincati to provide automatic node updates and reboot without interrupting services.
- **Dual rootfs**: executes dual rootfs for active/standby switchovers, to ensure integrity and security during system running.

Application Scenarios

openEuler introduces the NestOS-Kubernetes-Deployer tool to resolve problems such as inconsistent and repeated O&M operations across stacks and platforms. These problems are typically caused by decoupling of containers from underlying environments when using container and container orchestration technologies for rollout and O&M. NestOS is developed for containerized cloud applications and ensures consistency between services and the base OS.

7 Enhanced Features

SysCare

SysCare is a system-level hotfix software that provides security patches and hot fixing for OSs. It can fix system errors without restarting hosts. SysCare combines kernel-mode and user-mode hot patching to take over system repair, freeing up valuable time for users to focus on core services. Looking ahead, SysCare will introduce live OS update capabilities, further improving O&M efficiency.

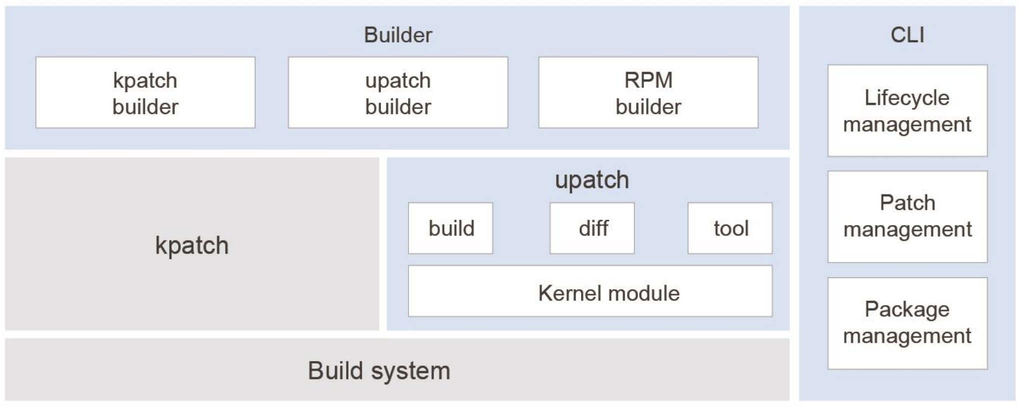
Feature Description

- Hot patch creation

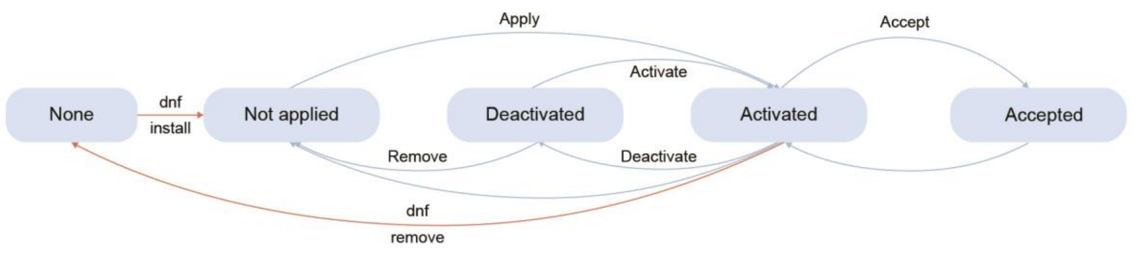
  Hot patch RPM packages can be generated by providing the paths of the source RPM package, debuginfo RPM package, and patches to be applied, eliminating the need to modify the software source code.
- Patch lifecycle management

SysCare simplifies patch lifecycle management, offering a complete and user-friendly solution. With a single command, users can efficiently manage hot patches, saving time and effort. SysCare leverages the RPM system to build hot patches with complete dependencies. This allows for easy integration into the software repository and simplifies distribution, installation, update, and uninstallation of hot patches.

SysCare architecture:



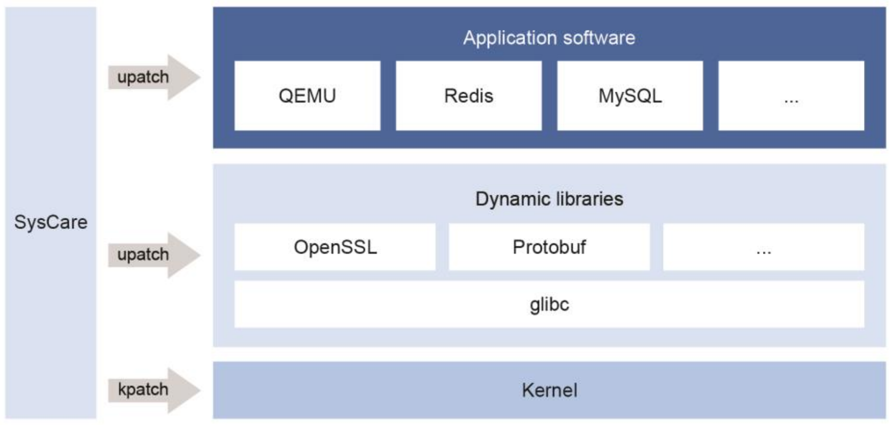
Lifecycle of a hot patch:



- **Kernel-mode and user-mode hot patch integration**

  By utilizing the upatch and kpatch technologies, SysCare delivers seamless hot fixing for the entire software stack from applications and dynamic libraries to the kernel, eliminating the need for disruptive downtime.

  Application scope of hot patches:



New features

SysCare preserves and restores accepted hot patches after reboot, maintaining their original activation order.

Constraints

- Only available for 64-bit OSs
- Only available for ELF files, with no support for interpreted languages or pure assembly modification
- Only available for the GCC and G++ compilers, not supporting cross compilation
- TLS variable modification not supported
- LTO not supported
- Compiler configuration interface for upatch-build only and not for syscare-build

Application Scenarios

Scenario 1: quick fix of common vulnerabilities and exposures (CVEs)

Scenario 2: temporary location for live-network issues

NRI Plugin Support of iSula

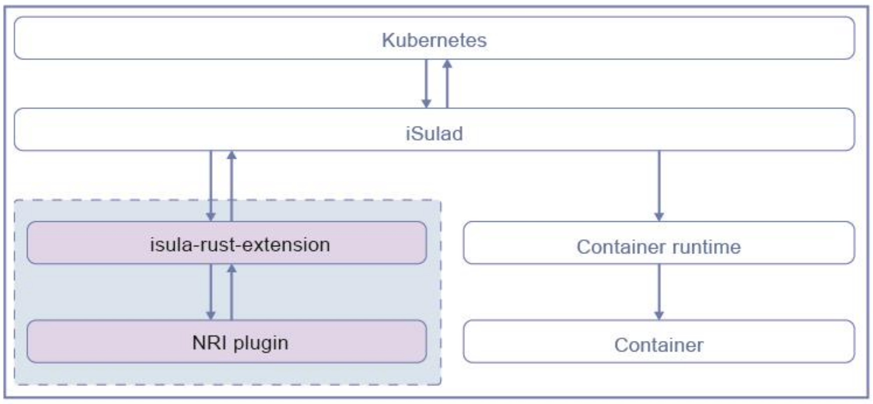
Node Resource Interface (NRI) is a public interface for controlling node resources and provides a generic framework for pluggable extensions for CRI-compatible container runtimes. It offers a fundamental mechanism for extensions to track container states and make limited modifications to their configurations. This allows users to insert custom logic into OCI-compatible runtimes, enabling controlled changes to containers or performing additional operations outside the scope of OCI at certain points in the container lifecycle. The newly added support for NRI plugins of iSulad reduces costs for container resource management and maintenance, eliminates scheduling delays, and ensures information consistency in Kubernetes environments.

Feature Description

An NRI plugin establishes a connection with iSulad through the NRI runtime service started within the isula-rust-extension component. This connection enables the plugin to subscribe to both pod and container lifecycle events.

- For pods, subscribable events include creation, stopping, and removal.
- For containers, subscribable events include creation, post-creation, starting, post-start, updating, post-update, stopping, and removal.

Upon receiving a CRI request from Kubernetes, iSulad relays this request to all NRI plugins subscribed to the corresponding lifecycle events. The request received by an NRI plugin includes metadata and resource information for the relevant pod or container. The plugin can then adjust the resource configuration of the pod or container as required. Finally, the NRI plugin communicates the updated configuration to iSulad, which in turn relays it to the container runtime, making the updated configuration effective.



Constraints

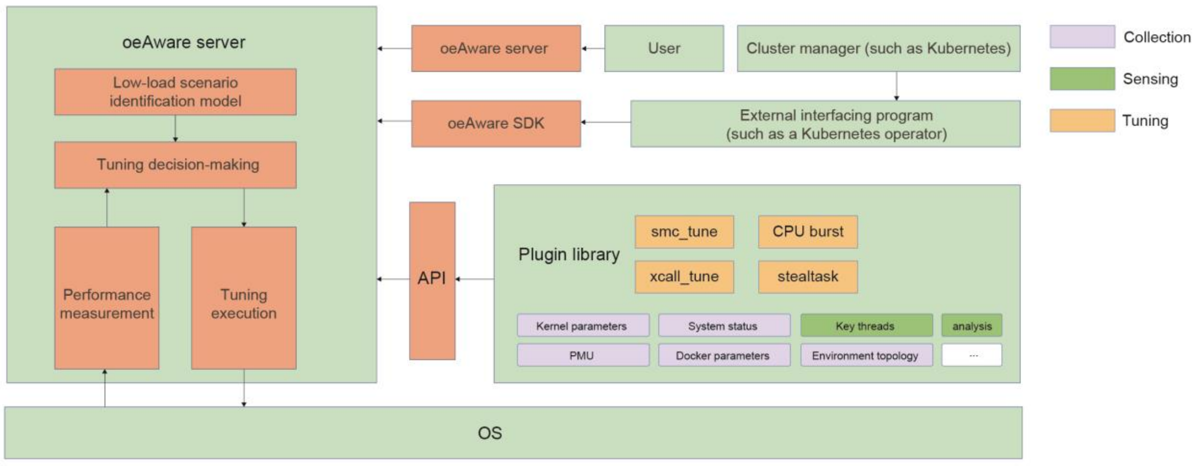
iSulad implements NRI within Kubernetes CRI V1, with support for NRI API version 0.6.1 only.

Application Scenarios

NRI support of iSulad can be utilized for resource management within Kubernetes environments. By implementing NRI plugins, users gain the capability to subscribe to container lifecycle events, enabling real-time tracking of container resource status. Based on predefined resource management logic, these plugins can then dynamically modify container configurations within the scope permitted by the NRI API. For instance, NRI facilitates the development of standardized plugins to address resource utilization and priority scheduling challenges inherent in scenarios where online and offline services are deployed together.

oeAware with Enhanced Information Collection and Tuning Plugins

oeAware is a framework that provides low-load collection, sensing, and tuning upon detecting defined system behaviors on openEuler. The framework divides the tuning process into three layers: collection, sensing, and tuning. Each layer is associated through subscription and developed as plugins, overcoming the limitations of traditional tuning techniques that run independently and are statically enabled or disabled.



Feature Description

Every oeAware plugin is a dynamic library that utilizes oeAware interfaces. The plugins comprise multiple instances that each contains several topics and deliver collection or sensing results to other plugins or external applications for tuning and analysis purposes. The framework consists of the following components:

- The SDK enables subscription to plugin topics, with a callback function handling data from oeAware. This allows external applications to create tailored functionalities, such as cross-cluster information collection or local node analysis.
- The Performance monitoring unit (PMU) information collection plugin gathers performance records from the system PMU.
- The Docker information collection plugin retrieves specific parameter details about the Docker environment.
- The system information collection plugin captures kernel parameters, thread details, and resource information (CPU, memory, I/O, network) from the current environment.
- The thread sensing plugin monitors key information about threads.
- The evaluation plugin examines system NUMA and network information during service operations, suggesting optimal tuning methods.
- The system tuning plugins comprise stealtask for enhanced CPU tuning, smc_tune which leverages shared memory communication in the kernel space to boost network throughput and reduce latency, and xcall_tune which bypasses non-essential code paths to minimize system call processing overhead.
- The Docker tuning plugin addresses CPU performance issues during sudden load spikes by utilizing the CPU burst feature.

Constraints

- smc_tune: SMC acceleration must be enabled before the server-client connection is established. This feature is most effective in scenarios with numerous persistent connections.
- Docker tuning is not compatible with Kubernetes containers.
- xcall_tune: The **FAST_SYSCALL** kernel configuration option must be activated.

Application Scenarios

The oeAware plugins offer an excellent option to boost CPU performance for a range of demanding workloads and across diverse scenarios.

stealtask boosts CPU utilization and prevents idle CPU cycles. Similarly, CPU burst is tailored for high-load container environments like Doris, resolving CPU bottlenecks.

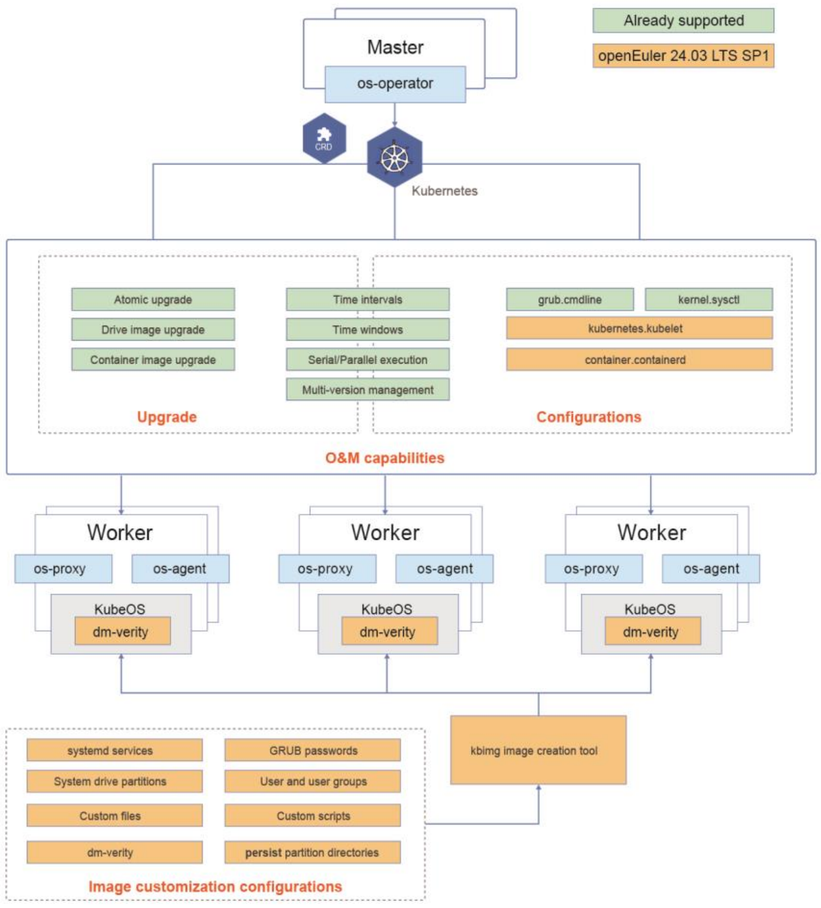
xcall_tune is designed for applications with substantial system call overhead, providing code paths that bypass non-critical processes to improve system call handling and reduce overheads. However, this approach may compromise maintenance and security capabilities.

For environments requiring high throughput and low latency, like HPC and cloud platforms, smc_tune leverages direct memory access (DMA) to significantly reduce CPU load and accelerate interactive workloads.

KubeOS

KubeOS is a lightweight and secure OS designed for cloud-native environments. It simplifies O&M by providing unified tools for Kubernetes-based systems. KubeOS is specifically designed for running containers. It features a read-only root directory, includes only the essential components for the container runtime, and utilizes dm-verity security hardening. This minimizes vulnerabilities and attack surfaces while improving resource utilization and boot speed. KubeOS can leverage native Kubernetes declarative APIs to unify the upgrade, configuration, and maintenance of worker node OSs within a cluster. This approach simplifies cloud-native operations, addresses challenges associated with OS version fragmentation across cluster nodes, and provides a unified solution for OS management.

Feature Description



KubeOS introduces enhanced configuration capabilities, image customization features, and root file system (rootfs) integrity protection using dm-verity, as shown in the figure. Through a centralized management platform, KubeOS enables unified configuration of cluster parameters in the **limits.conf** file and cluster components like containerd and kubelet.

KubeOS provides comprehensive system image customization options, allowing users to configure systemd services, GRUB passwords, system drive partitions, users and user groups, files, scripts, and **persist** partition directories.

For static integrity protection, KubeOS activates dm-verity during VM image creation to ensure rootfs integrity. It also supports upgrades and configuration when dm-verity is enabled.

Application Scenarios

KubeOS provides unified management of cluster and node components like containerd and kubelet to simplify complex cluster setup and ensure consistency, streamlining maintenance and reducing manual operations from days to hours. Its image customization tool helps users develop tailored cloud-native OS images. The boot integrity verification ensured trusted and immutable infrastructure, ensuring each system boot satisfies prescribed requirements, preventing privileged attacks and unauthorized changes to system settings (namely rootkit attacks) and creating a secure cloud-native OS.

IMA

The IMA is an open source Linux technology widely used for file integrity protection in real-world applications. It performs integrity checks on system programs to prevent tampering and ensure only authenticated (signed or HMAC-verified) files are executed via an allowlist.

Applications running on Linux can be categorized into two types:

- **Binary executables**: Programs in the ELF format can be directly executed by **exec** or **mmap** system calls. Through hook functions in **exec** and **mmap** system calls, the IMA triggers measurement or verification processes, ensuring integrity protection.
- **Interpreter-based applications**: Programs indirectly executed via interpreters, such as scripts run by Bash, Python, and Lua interpreters and Java programs executed by the JVM.

The current IMA fails to protect interpreter-based applications, as these applications are typically loaded and parsed by interpreters through **read** system calls. The IMA cannot differentiate them from other mutable files, such as configuration or temporary files. As a result, enabling the IMA for **read** system calls inadvertently includes these mutable files in the protection scope. Mutable files lack pre-generated measurement baselines or verification credentials, causing integrity check failures.

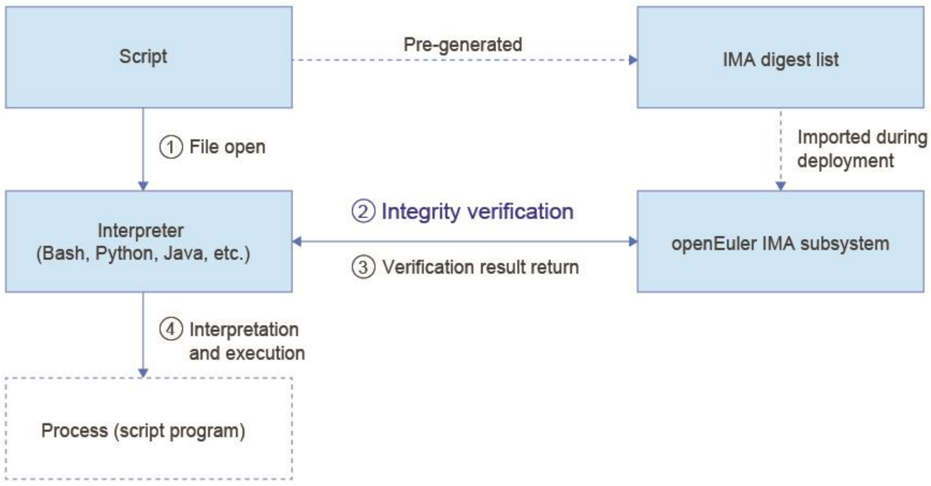
To address this limitation, openEuler enhances the IMA feature to significantly improve integrity protection for interpreter-based applications.

Feature Description

This feature extends the existing Linux permission mechanism to cover programs executed directly via **exec** (such as **./test.sh**) and those executed indirectly through interpreters (like **bash ./test.sh**), ensuring the programs follow the same permission check process. The specific implementation includes:

- openEuler 24.03 LTS SP1 supports the **AT_CHECK** flag in the **execveat** system call, enabling executable permission checks on files and returning results without actual execution.
- Components like Bash and JDK can now invoke **execveat** with the **CHECK** flag for executable permission checks on script or Java files, ensuring programs can only proceed if the check succeeds.

The enhanced IMA mechanism enables integrity protection for interpreter-based applications. The interpreter triggers executable permission checks on script programs via **execveat** and indirectly activates the IMA for **exec** system calls. This approach ensures integrity protection for script programs and, by selectively enabling the IMA for **read** system calls, avoids overextending the protection.


Script
Pre-generated
IMA digest list
① File open
Imported during deployment
② Integrity verification
Interpreter (Bash, Python, Java, etc.)
openEuler IMA subsystem
③ Verification result return
④ Interpretation and execution
Process (script program)

Community developers or users can leverage this feature to extend support for additional interpreters or similar mechanisms.

Application Scenarios

An application allowlist can be implemented for high-security environments to allow only binary or script programs verified or measured by the IMA to be executed.

Heterogeneous RoT Support

Common attack methods often target the authenticity and integrity of information systems. Hardware RoTs have become a standard method for protecting critical system components, enabling the system to measure and verify integrity. When tampering or counterfeiting is detected, the system triggers alerts or blocks the activity.

The prevailing protection approach uses the trusted platform module (TPM) as the RoT, combined with the integrity measurement software stack to establish a system trust chain that ensures system authenticity and integrity. openEuler supports integrity measurement features such as measured boot, IMA measurement for files, and dynamic integrity measurement (DIM) for memory.

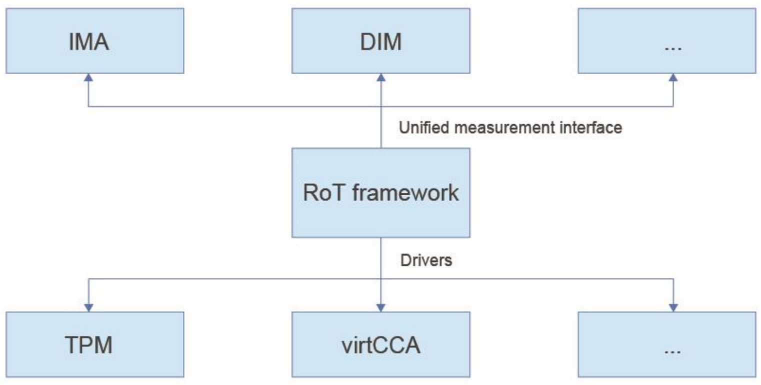
Recent advancements in trusted computing and confidential computing have produced diverse hardware RoTs and associated attestation systems. Examples include:

- Trusted Cryptography Module (TCM)
- Trusted Platform Control Module (TPCM)
- Intel Trust Domain Extensions (TDX)
- Arm Confidential Compute Architecture (CCA)
- Virtualized Arm Confidential Compute Architecture (virtCCA)

To enable openEuler's integrity measurement software stack across different hardware RoTs, openEuler 24.03 LTS SP1 introduces an RoT framework for heterogeneous instances.

Feature Description

The RoT framework used by openEuler 24.03 LTS SP1 offers a unified measurement interface to the upper-layer integrity protection software stack. Deployed within the kernel integrity subsystem, the framework supports multiple RoT drivers and expands integrity measurement beyond the TPM to include heterogeneous RoTs.



Application Scenarios

Users can enable the IMA feature in TPM and virtCCA-based environments and verify the integrity of IMA measurement results using RoTs. This framework can also help community developers and hardware vendors in their own projects or infrastructure by supporting other integrity measurement features and hardware.

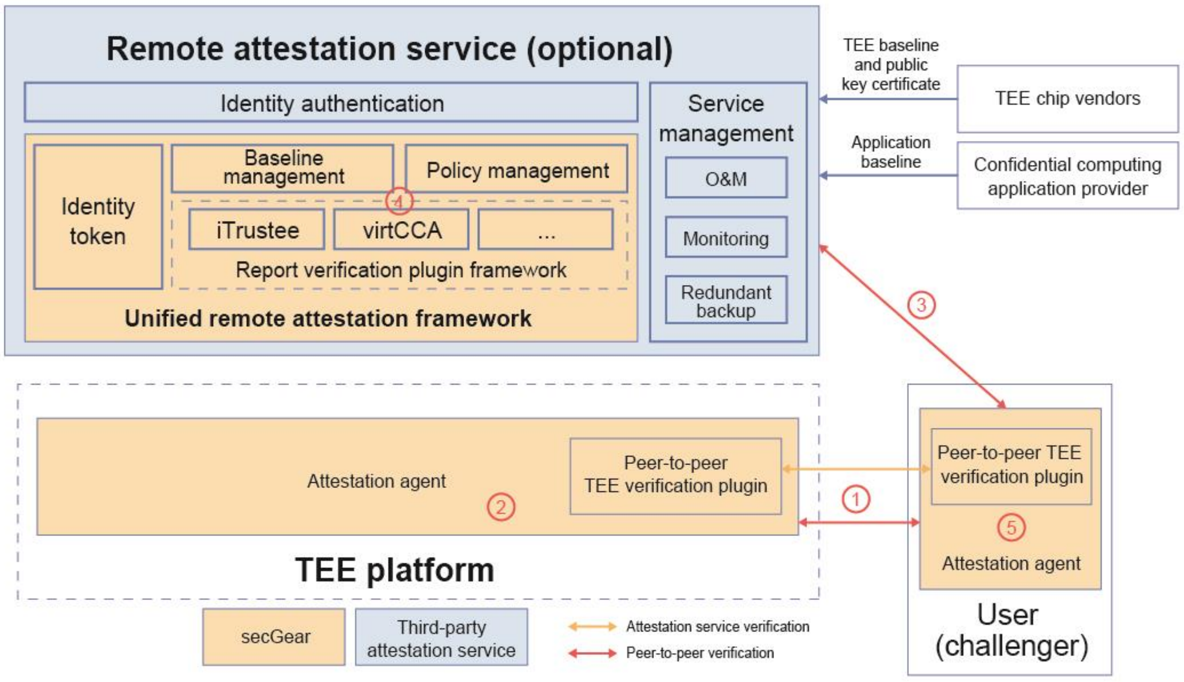
Enhanced secGear

The unified remote attestation framework of secGear addresses the key components related to remote attestation in confidential computing, abstracting away the differences between different Trusted Execution Environments (TEEs). It provides two components: attestation agent and attestation service. The agent is integrated by users to obtain attestation reports and connect to the attestation service. The service can be deployed independently and supports the verification of iTrustee and virtCCA remote attestation reports.

Feature Description

The unified remote attestation framework focuses on confidential computing functionalities, while service deployment and operation capabilities are provided by third-party deployment services. The key features of the unified remote attestation framework are as follows:

- **Report verification plugin framework**: Supports runtime compatibility with attestation report verification for different TEE platforms, such as iTrustee, virtCCA, and CCA. It also supports the extension of new TEE report verification plugins.
- **Certificate baseline management**: Supports the management of baseline values of Trusted Computing Bases (TCB) and Trusted Applications (TA) as well as public key certificates for different TEE types. Centralized deployment on the server ensures transparency for users.
- **Policy management**: Provides default policies for ease of use and customizable policies for flexibility.
- **Identity token**: Issues identity tokens for different TEEs, endorsed by a third party for mutual authentication between different TEE types.
- **Attestation agent**: Supports connection to attestation services/peer-to-peer attestation, compatible with TEE report retrieval and identity token verification. It is easy to integrate, allowing users to focus on their service logic.


Remote attestation service (optional)
Identity authentication
Identity token
Baseline management
Policy management
iTrustee
virtCCA
...
Report verification plugin framework
Unified remote attestation framework
Service management
O&M
Monitoring
Redundant backup
TEE baseline and public key certificate
TEE chip vendors
Application baseline
Confidential computing application provider
Attestation agent
Peer-to-peer TEE verification plugin
TEE platform
Peer-to-peer TEE verification plugin
Attestation agent
User (challenger)
secGear
Third-party attestation service
Attestation service verification
Peer-to-peer verification
1
2
3
4
5

Two modes are supported depending on the usage scenario: peer-to-peer verification and attestation service verification.

Attestation service verification process:

1. The user (regular node or TEE) initiates a challenge to the TEE platform.
2. The TEE platform obtains the TEE attestation report through the attestation agent and returns it to the user.
3. The user-side attestation agent forwards the report to the remote attestation service.
4. The remote attestation service verifies the report and returns an identity token in a unified format endorsed by a third party.
5. The attestation agent verifies the identity token and parses the attestation report verification result.

Peer-to-peer verification process (without the attestation service):

1. The user initiates a challenge to the TEE platform, which then returns the attestation report to the user.
2. The user uses a local peer-to-peer TEE verification plugin to verify the report.

Note: The attestation agents used for peer-to-peer verification and attestation service verification are different. During compilation, the compilation options determine whether to compile attestation agents for the attestation service mode or peer-to-peer mode.

Application Scenarios

In scenarios like finance and AI, where confidential computing is used to protect the security of privacy data during runtime, remote attestation is a technical means to verify the legitimacy of the confidential computing environment and applications. secGear provides components that are easy to integrate and deploy, helping users quickly enable confidential computing remote attestation capabilities.

gala-anteater for Minute-Level Container Interference Detection

gala-anteater is an AI-powered exception detection platform for gray faults in the OS. Integrating various exception detection algorithms, it achieves system-level fault detection and reporting through automated model pre-training, online incremental learning, and model updates.

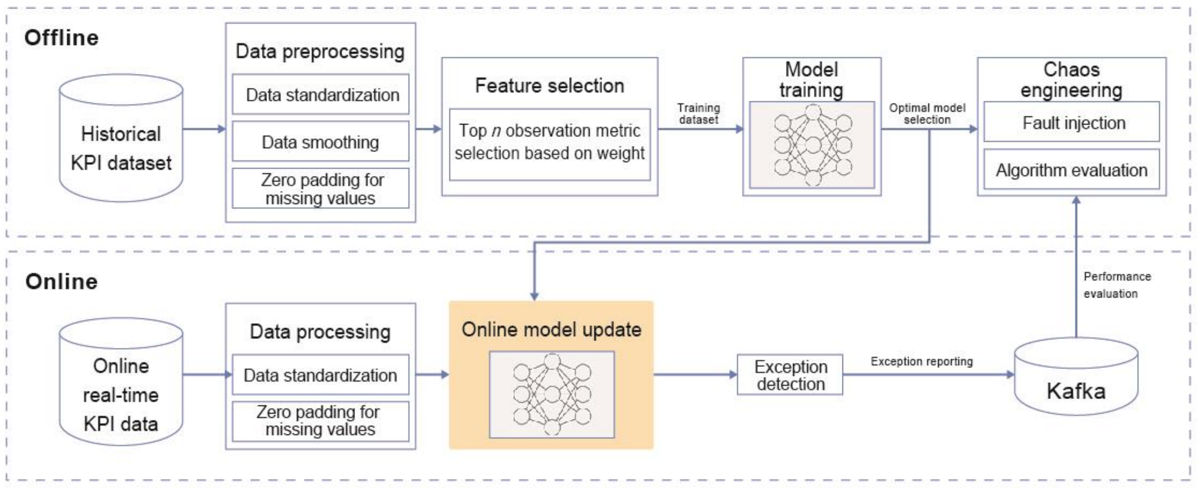
In high-density online container deployment scenarios, the presence of disorderly resource contention can lead to inter-container interference. gala-anteater enables minute-level identification of interference sources (CPU or I/O), aiding O&M personnel in swiftly tracing and resolving issues to ensure service QoS.

Feature Description

gala-anteater leverages a combination of offline and online learning techniques to facilitate offline model training and online updates, ultimately enabling real-time online exception detection.

Offline: Initially, historical KPI datasets undergo preprocessing and feature selection to generate a training set. This set is then used to train and optimize an unsupervised neural network model (such as a variational autoencoder). Finally, a manually labeled test set aids in selecting the optimal model.

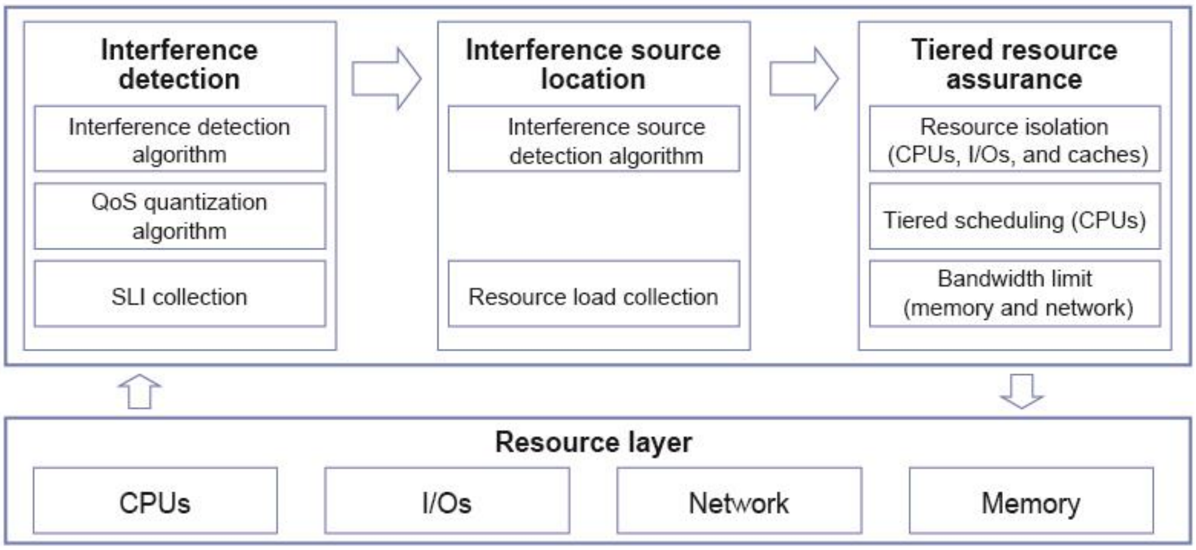
Online: The trained model is deployed online, where it undergoes further training and parameter tuning using real-time data. This continuously refined model then performs real-time exception detection within the online environment.



Application Scenarios

gala-anteater supports both application-level and system-level exception detection and reporting.

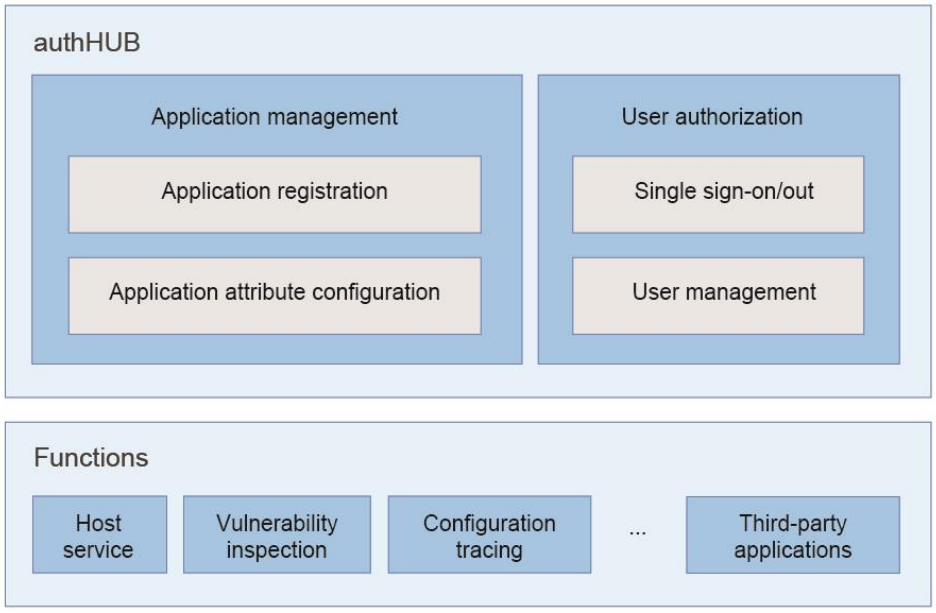
In actual application scenarios, it can be integrated with gala-gopher and Rubik to achieve real-time interference detection, location, and self-healing. gala-anteater leverages real-time resource-level metrics collected by gala-gopher for interference detection and location. The reported detection results are then used by Rubik to implement tiered resource assurance, enabling rapid recovery from interference.


Interference detection
Interference detection algorithm
QoS quantization algorithm
SLI collection
Interference source location
Interference source detection algorithm
Resource load collection
Tiered resource assurance
Resource isolation (CPUs, I/Os, and caches)
Tiered scheduling (CPUs)
Bandwidth limit (memory and network)
Resource layer
CPUs
I/Os
Network
Memory

A-Ops Integration with authHub for Unified User Authentication

authHub is a unified user authentication platform built on the OAuth 2.0 protocol. A-Ops can now utilize authHub to manage application registration, enabling seamless user authentication across multiple platforms.

Feature Description



Core functionalities of authHub include:

- **Application management**: Deployed applications can be registered with and configured on authHub to provide access to their features.

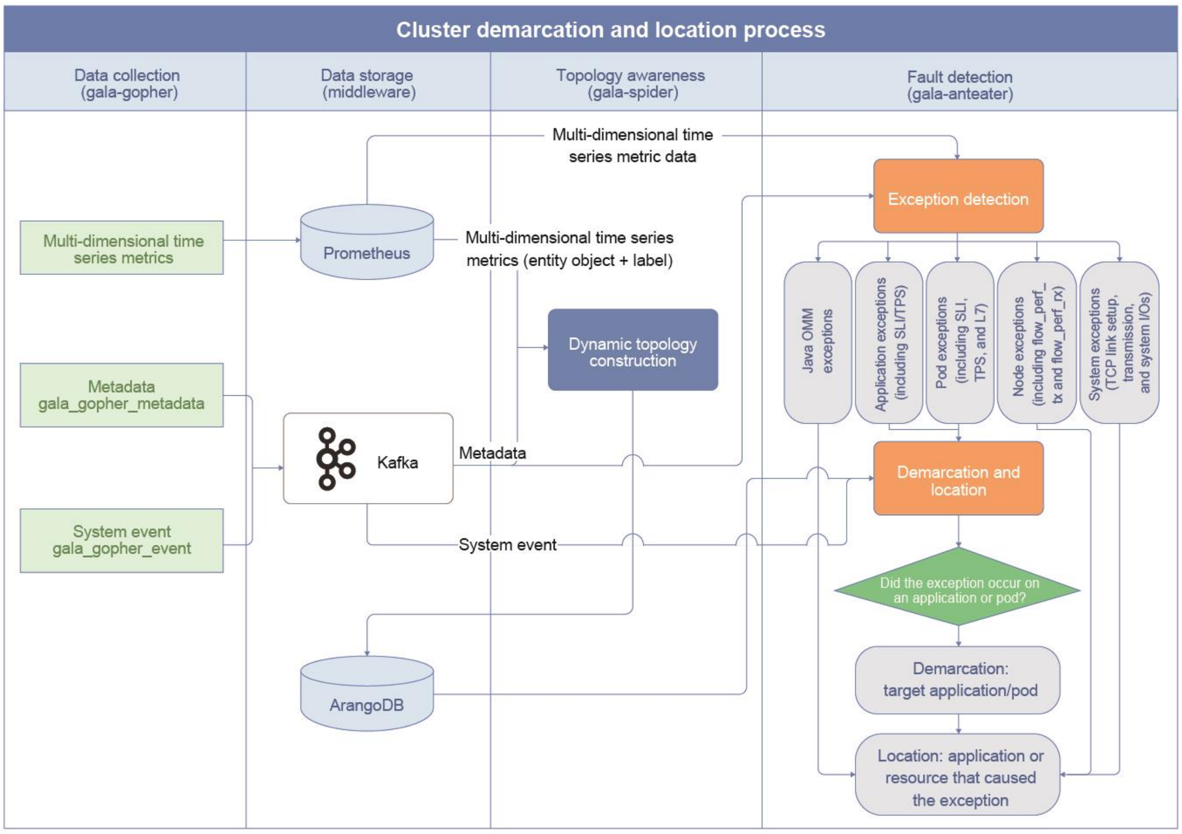
- **User authentication**: Applications managed by authHub support single sign-on (SSO) and single sign-out processes.

Application Scenarios

A-Ops enables third-party authentication systems to connect to authHub for improved application scalability, central management, and consistent control over service access permissions.

Minute-level Demarcation and Location of Microservice Performance Problems (TCP, I/Os, and Scheduling)

Feature Description


Cluster demarcation and location process
Data collection (gala-gopher)
Data storage (middleware)
Topology awareness (gala-spider)
Fault detection (gala-anteater)
Multi-dimensional time series metric data
Multi-dimensional time series metrics
Prometheus
Multi-dimensional time series metrics (entity object + label)
Exception detection
Java OMM exceptions
Application exceptions (including SLI/TPS)
Pod exceptions (including SLI, TPS, and L7)
Node exceptions (including flow_perf_tx and flow_perf_rx)
System exceptions (TCP link setup, transmission, and system I/Os)
Dynamic topology construction
Metadata gala_gopher_metadata
Kafka
Metadata
Demarcation and location
System event gala_gopher_event
System event
Did the exception occur on an application or pod?
ArangoDB
Demarcation: target application/pod
Location: application or resource that caused the exception

gala-gopher, gala-spider, and gala-anteater implement a topological root cause analysis approach to facilitate fault detection and root cause location in large-scale clusters. In openEuler 24.03 LTS SP1, fine-grained capabilities have been introduced for cloud-native environments based on Layer 7 protocols like HTTPS, PostgreSQL, and MySQL, enabling O&M teams to quickly locate the source of faults, thus enhancing system stability and reliability. The following features are available:

- **Metric collection**: gala-gopher uses eBPF to collect and report network and I/O metrics.
- **Cluster topology**: gala-spider receives data reported by gala-gopher and constructs a container- and process-level call relationship topology.
- **Fault detection**: gala-anteater classifies the reported metrics based on the fault detection model to determine whether an exception has occurred in the system.

- **Root cause location**: gala-anteater locates the root cause node of the exception based on node exception and topology information.

Application Scenarios

Minute-level demarcation and location of application problems are suited to the following scenarios:

- **Cloud Kubernetes pods**: Enterprises with cloud environments can use gala-gopher to collect metrics of Kubernetes containers and processes, gala-spider to construct a call relationship topology, and gala-anteater to detect faults and locate root causes based on their requirements and IT resource status.
- **Bare metal deployments**: Enterprises with bare metal deployments can use gala-gopher to collect metrics of processes and containers, gala-spider to construct a call relationship topology, and gala-anteater to detect faults and locate root causes based on their requirements and IT resource status.
- **Large-scale VMs**: Enterprises with large-scale VMs can use gala-gopher to collect metrics of VMs, gala-spider to construct a call relationship topology, and gala-anteater to detect VM faults and locate root causes based on their requirements and IT resource status.

utsudo

sudo is one of the commonly used utilities for Unix-like and Linux OSs. It enables users to run specific commands with the privileges of the super user. utsudo is developed to address issues of security and reliability common in sudo.

utsudo uses Rust to reconstruct sudo to deliver more efficient, secure, and flexible privilege escalation. It includes modules such as the common utility, overall framework, and function plugins.

Feature Description

Basic features

- **Access control**: Limits the commands that can be executed by users, and specifies the required authentication method.
- **Audit log**: Records and traces all commands and tasks executed by each user.
- **Temporary privilege escalation**: Allows common users to temporarily escalate to a super user for executing privileged commands or tasks.
- **Flexible configuration**: Allows users to set arguments such as command aliases, environment variables, and execution parameters to meet system requirements.

Enhanced features

utsudo 0.0.2 comes with openEuler 24.03 LTS SP1 to provide the following features:

- **Privilege escalation**: Escalates the privilege of a process run by a common user to the **root** privilege.
- **Plugin loading**: Parses plugin configuration files and dynamically loads plugin libraries.

Application Scenarios

utsudo reduces security risks by enabling administrators to better manage user privileges and authorization, and prevent unauthorized privileged operations.

It is suitable for management and maintenance, user permission control, and multi-user environments.

utshell

utshell is a new shell that introduces new features and inherits the usability of Bash. It enables interaction through command lines, such as responding to user operations to execute commands and providing feedback, and can execute automated scripts to facilitate O&M.

Feature Description

Basic features

- **Command execution**: Runs and sends return values from commands executed on user machines.
- **Batch processing**: Automates task execution using scripts.
- **Job control**: Executes, manages, and controls multiple user commands as background jobs concurrently.
- **Historical records**: Records the commands entered by users.
- **Command aliases**: Allows users to create aliases for commands to customize their operations.

Enhanced features

utshell 0.5 runs on openEuler 24.03 LTS SP1 to perform the following operations:

- Parses shell scripts.
- Runs third-party commands.

Application Scenarios

utshell is well suited to conventional server, cloud-native environments, and attended or unattended production sites where automated O&M scripts are executed, as well as the demands of individual users.

GCC for openEuler

The baseline version of GCC for openEuler has been upgraded from open source GCC 10.3 to GCC 12.3, supporting features such as automatic feedback-directed optimization (AutoFDO), software and hardware collaboration, memory optimization, Scalable Vector Extension (SVE), and vectorized math libraries.

- The default language of GCC for openEuler has been upgraded from C14/C++14 to C17/C++17, enabling GCC for openEuler to support more hardware features like Armv9-A and x86 AVX512-FP16.

Item	GCC 10.3.0	GCC 11.3.0	GCC 12.3.0
Release date	2021-04-08	2022-04-21	2023-05-08
C standard	C17 by default C2x supported	C17 by default C2x supported	C17 by default C2x supported

Item	GCC 10.3.0	GCC 11.3.0	GCC 12.3.0
C++ standard	C++14 by default C++17 supported C++2a experimental optimization C++20 partially supported	C++17 supported C++2a experimental optimization C++20 partially supported	C++17 supported C++2a experimental optimization C++20 partially supported
New architecture features	Armv8.6-a (BFloat16 Extension/Matrix Multiply Extension) SVE2 Cortex-A77 Cortex-A76AE Cortex-A65 Cortex-A65AE Cortex-A34	Armv8.6-a, +bf16, +i8mm Armv8.6-r Cortex-A78 Cortex-A78AE Cortex-A78C Cortex-X1	Armv8.7-a, +ls64 atomic load and store Armv8.8-a, +mop, accelerate memory operations Armv9-a Ampere-1 Cortex-A710 Cortex-X2 AVX512-FP16 SSE2-FP16

- GCC for openEuler supports structure optimization and instruction selection optimization, leveraging Arm hardware features to improve system running efficiency. In the benchmark tests such as SPEC CPU 2017, GCC for openEuler has proven to deliver higher performance than GCC 10.3 of the upstream community.
- Further, it fuels AutoFDO to improve the performance of MySQL databases at the application layer.

Feature Description

- **SVE vectorization**: Significantly improves program running performance for Arm-based machines that support SVE instructions.
- **Memory layout**: Rearranges the structure members so that frequently accessed members are placed in continuous memory locations, boosting the cache hit ratio and enhancing program performance.
- **SLP transpose optimization**: Improves the analysis of loops with consecutive memory reads during loop splitting, and adds analysis to transpose grouped stores in the superword level parallelism (SLP) vectorization stage.
- **Redundant member elimination**: Eliminates structure members that are never read and deletes redundant write statements, which in turn reduces the memory footprint of the structure and alleviates subsequent bandwidth pressure, while improving performance.
- **Array comparison**: Implements parallel comparison of array elements to improve execution efficiency.
- **Arm instruction optimization**: Simplifies the pipeline of ccmp instructions for a wide range of deployments.
- **IF statement optimization**: Splits and optimizes the IF statement block to improve constant propagation within a program.

- **SLP vectorization**: Enhances SLP to cover more vectorization scenarios and improve performance.
- **AutoFDO**: Uses perf to collect and parse program information and optimizes feedback across the compilation and binary phases, boosting mainstream applications such as MySQL databases.

Application Scenarios

In general-purpose computing, GCC for openEuler showed a 20% performance gain over GCC 10.3 in the SPEC CPU 2017 benchmark.

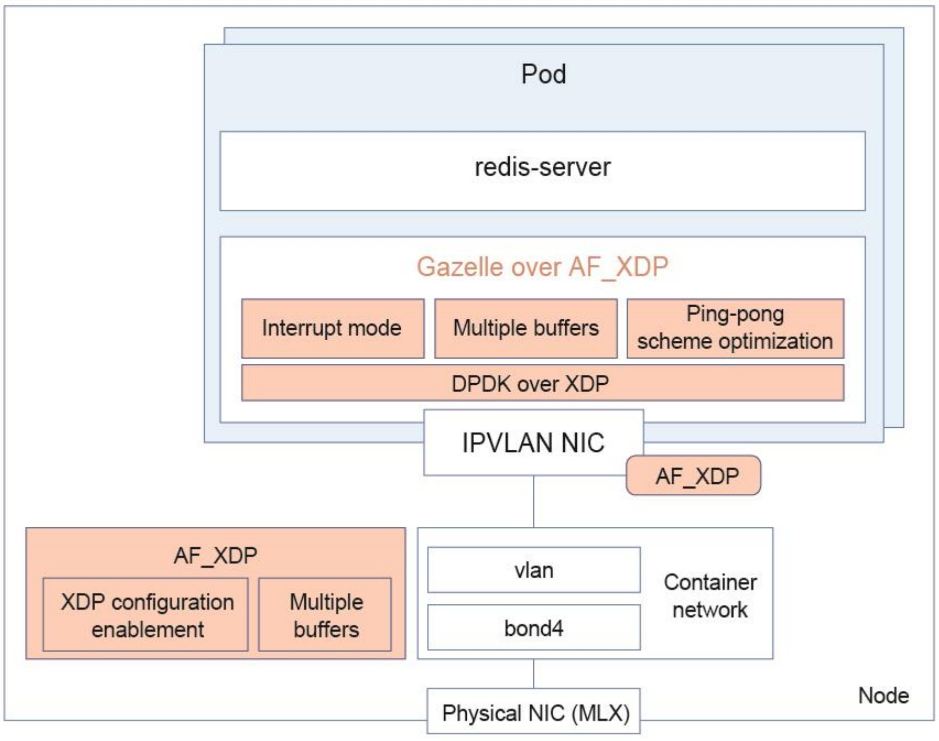
In other scenarios, MySQL performance increases by 15% with AutoFDO enabled, while UnixBench performance is boosted by over 3% with kernel-mode PGO.

Gazelle

Gazelle is a high-performance user-mode protocol stack. It directly reads and writes NIC packets in user mode based on the Data Plane Development Kit (DPDK), transmits the packets through shared hugepage memory, and uses the LwIP protocol stack, thereby greatly improving the network I/O throughput of applications and accelerating the network for databases. With Gazelle, high performance and universality can be achieved at the same time. In openEuler 24.03 LTS SP1, Gazelle introduces eXpress Data Path (XDP) deployment mode for container scenarios and TPC-C support for the openGauss database, further enhancing the user-mode protocol stack.

Feature Description

XDP network deployment mode:



- **High performance (ultra-lightweight)**: High-performance lightweight protocol stack capabilities are implemented based on DPDK and LwIP.
- **Ultimate performance**: A highly linearizable concurrent protocol stack is implemented based on technologies such as regional hugepage splitting, dynamic core binding, and full-path zero-copy.
- **Hardware acceleration**: TCP Segmentation Offload (TSO), checksum (CSUM) offload, Generic Receive Offload (GRO), and other offload technologies streamline the vertical acceleration of hardware and software.
- **Universality (POSIX compatibility)**: Full compatibility with POSIX APIs eliminates the need to modify applications. The recvfrom and sendto interfaces of UDP are supported.
- **General networking model**: Adaptive scheduling of the networking model is implemented based on mechanisms such as fd router and wake-up proxy. The UDP multi-node multicast model meets the requirements of any network application scenario.
- **Usability (plug-and-play)**: LD_PRELOAD enables zero-cost deployment by removing the requirement for service adaptation.
- **Easy O&M (O&M tool)**: Complete O&M methods, such as traffic statistics, metric logs, and CLI commands, are provided.

New features

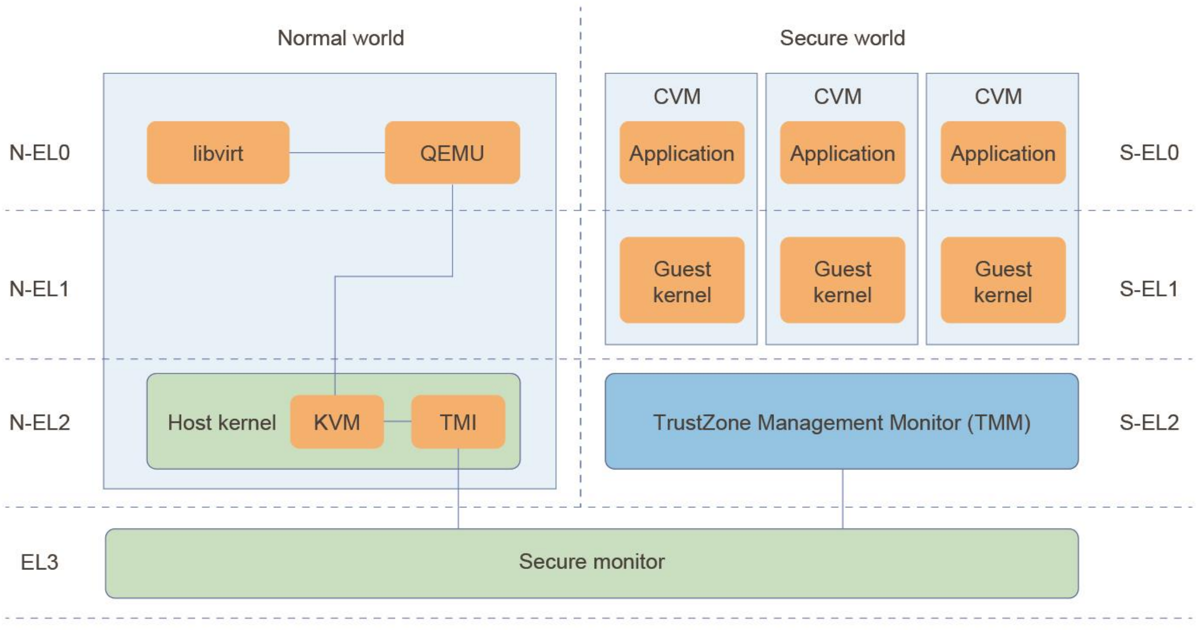
- Gazelle now supports XDP deployment on L2-mode NICs using IPVLAN.
- Interrupt mode is introduced, which reduces LStack CPU usage in no-traffic or low-traffic scenarios.
- Networks of the ping-pong scheme are optimized to improve packet transmission during ping-pong operations.
- Support for single-node and single-active/standby TPC-C testing is added for the openGauss database.

Application Scenarios

Gazelle is ideal for applications where network I/O is a performance bottleneck. It delivers significant performance improvements for databases such as Redis and MySQL.

virtCCA-based VMs

virtCCA-based confidential VMs, built on the Secure EL2 (S-EL2) of Kunpeng 920, allow regular VM software stacks to run securely within TEEs.



Based on the standard interface of the Arm Confidential Compute Architecture (CCA), openEuler builds a TEE virtualization management module upon the TrustZone firmware. This module supports memory isolation between confidential VMs, context and lifecycle management, and page table management, and enables seamless application migration to TEEs.

Constraints

For security purposes, hyper-threading needs to be disabled in the host BIOS.

Feature Description

- Device passthrough

  Device passthrough uses the PCIe protection controller (PCIPC) embedded in the PCIe root complex of Kunpeng 920. A selector is added to the PCIe bus to regulate communication between the processor and peripherals. Operating through the system memory management unit (SMMU), this selector controls both inbound and outbound traffic, safeguarding the entire data link.

  Device passthrough boasts excellent security isolation and performance enhancements for PCIe devices:

  - Security isolation

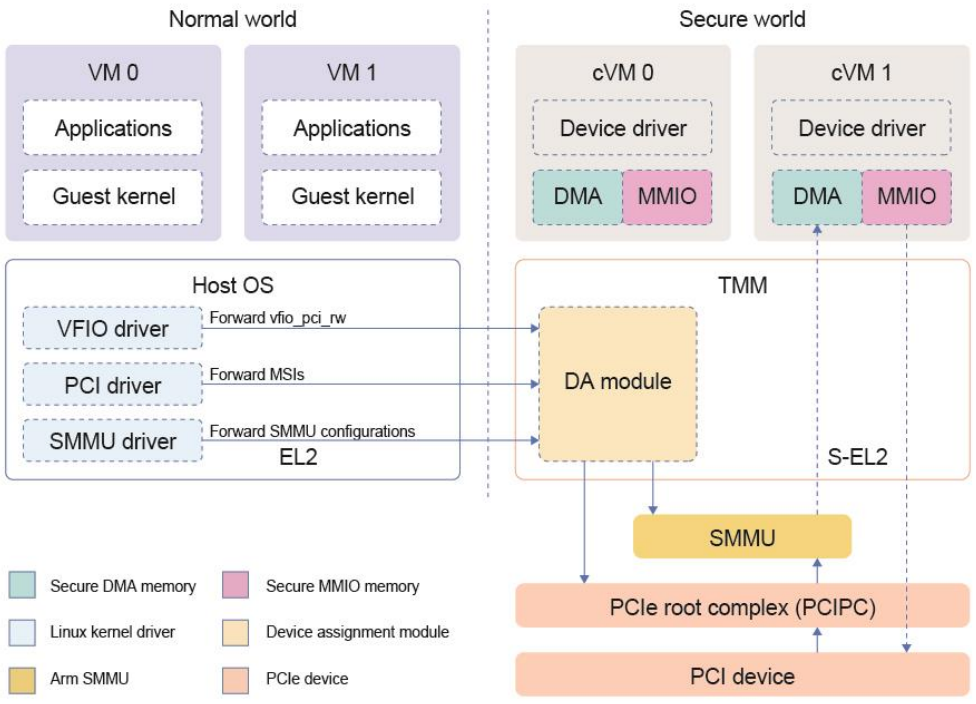
    The TEE manages device access permissions, preventing host software from accessing TEE devices.

  - High performance

    Confidential device passthrough negates performance loss on the data plane compared to traditional encryption and decryption solutions.

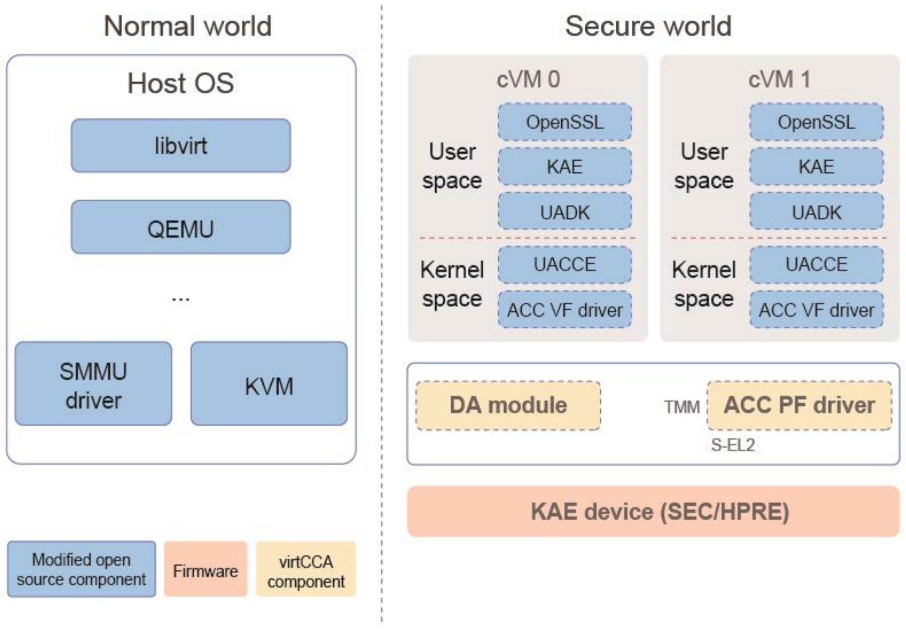
  - Ease of use

    Compatibility with existing open source OSs ensures that the kernel driver does not need to be modified.



- ShangMi (SM) algorithm acceleration

  Hardware-based acceleration for SM algorithms runs on the UADK user-mode accelerator framework to enhance SM algorithm performance and enable algorithm offloading within confidential VMs. It is powered by the Kunpeng processor and utilizes the Kunpeng Accelerator Engine (KAE) features in the TEE.



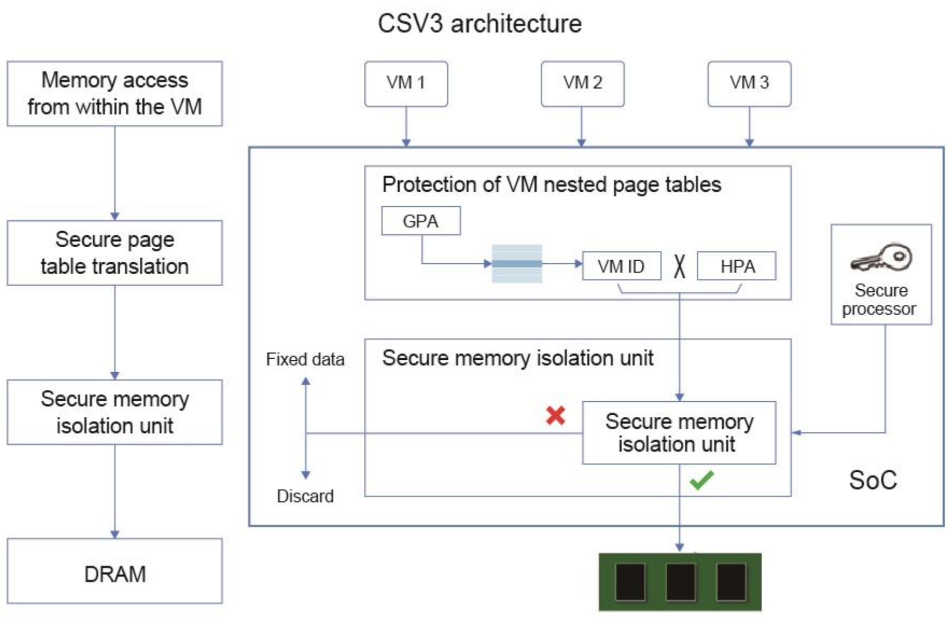
Application Scenarios

virtCCA-based VMs are suited to workloads from encrypted databases, secure cloud servers, confidential multi-party computation (MPC), trusted data circulation, and AI model data protection.

Hygon CSV3

CSV3 marks the third generation of Hygon's secure virtualization technology. With security features that far exceed its predecessors, CSV3 realizes multi-level data protection that implements VM data isolation within the CPU. This isolation prevents the host OS from accessing VM memory or modifying nested page tables, thereby ensuring strong data integrity and confidentiality.

Feature Description


CSV3 architecture
Memory access from within the VM
Secure page table translation
Secure memory isolation unit
DRAM
VM 1
VM 2
VM 3
Protection of VM nested page tables
GPA
VM ID
HPA
Secure processor
Fixed data
Discard
Secure memory isolation unit
Secure memory isolation unit
SoC

Secure memory isolation unit

The secure memory isolation unit forms the hardware basis for VM data integrity. A specialized hardware component within the CPU, the unit is positioned on the system bus path between the CPU and the memory controller to retrieve secure memory information for CSV3 VMs, such as physical memory addresses, VM IDs, and associated permissions. It validates all memory access requests from the CPU before granting permission, ensuring only verified requests receive access clearance.

When a CSV3 VM reads from or writes to memory, the page table translation unit converts the guest physical address (GPA) into a host physical address (HPA), and sends a memory access request (including read/write commands), the HPA, and the requester VM ID to the address bus.

If the secure memory isolation unit detects an incorrect VM ID during memory reads, it returns data in a fixed pattern. During memory writes, such an invalid request is discarded.

Secure processor

The secure memory isolation unit is the central hardware component in CSV3, safeguarding the integrity of VM memory. Its configuration must remain entirely secure and immune to modification by the host OS.

Hygon's secure processor is built into the SoC and independent from the CPU. It acts as the RoT of the SoC. Upon power-on, the processor verifies firmware integrity using an embedded signature verification key before the firmware is loaded and executed. Equipped with dedicated hardware resources within an isolated environment, the secure processor represents the highest level of security within the SoC, governing the overall security of the SoC. It maintains exclusive control over the secure memory isolation unit throughout the VM lifecycle, thus preventing the host OS from accessing or modifying unit configurations.

During VM startup, the host OS sends a request to the secure processor, which initializes the secure memory isolation unit. The secure processor firmware updates the secure memory isolation unit, and when the VM terminates, the secure processor clears all configurations of the unit. The secure processor validates all configuration requests from the host and rejects any that are invalid.

Protection of the virtual machine control block (VMCB)

The VMCB contains crucial control data, including the VM ID, VM page table base address, and VM register page base address. Access to this sensitive data by a host OS can cause tampering to VM memory data.

The secure processor creates the VMCB and places it under the protection scope of the secure memory isolation unit, which prevents the host OS from altering the contents of the VMCB.

To improve compatibility with the host OS software, CSV3 uses real and shadow VMCB pages. The host OS constructs the shadow VMCB page, populates it with control data, and sends it to the secure processor. The secure processor then generates the real VMCB page, copying non-critical control data from the shadow VMCB page and adding essential control information independently. The VM launches and operates using the real VMCB page, thus blocking any host OS attempts to hack the VMCB.

Application Scenarios

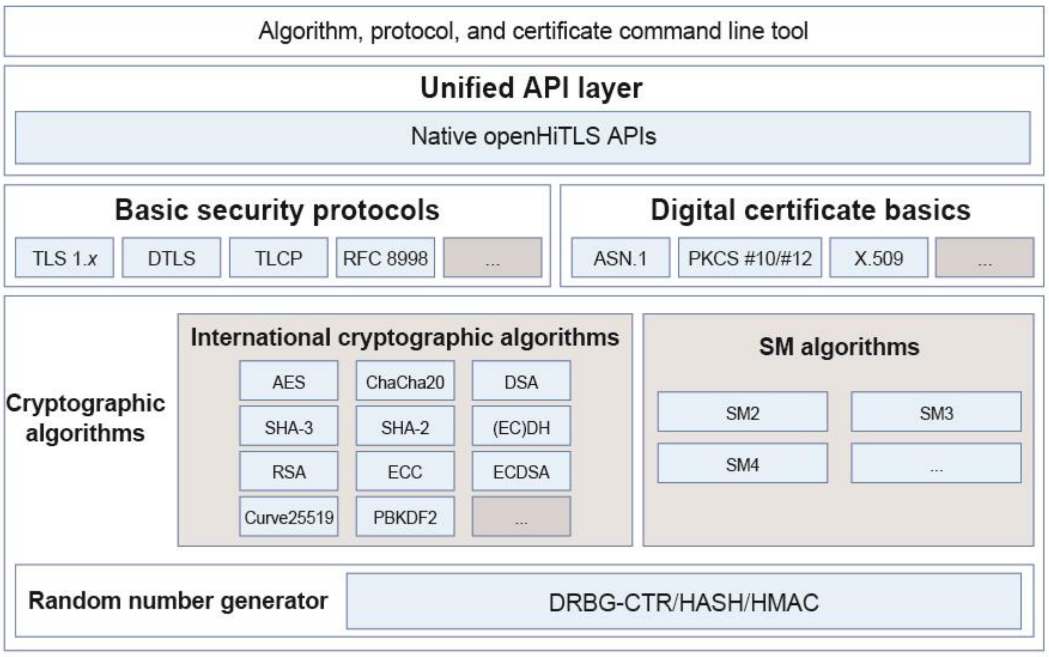
Hygon's CSV leverages secure virtualization and CPU hardware to protect data within the TEE. The technology implements the SM4 algorithm to ensure VM data confidentiality in memory. Through three evolutionary generations, Hygon has released CSV1, CSV2, and CSV3. CSV1 utilizes isolated TLB and cache resources of the C86 processor, while encrypting memory data via an SM4 engine on the memory controller. CSV2 enhances CSV1 by adding encryption for VM state information. CSV3 builds upon CSV2 by implementing hardware-based memory isolation to ensure VM data integrity.

VMs powered by CSV feature core security capabilities including boot measurement, application measurement, remote attestation, drive encryption, and key seal that are widely used across cloud computing, privacy computing, and other sectors.

For heterogeneous confidential computing, Hygon offers a comprehensive security solution integrating CSV and the deep-learning computing unit (DCU). This solution features confidential VRAM to prevent unauthorized access, encrypted data transmission to ensure link security, and remote attestation to verify DCU identity. The solution is positioned for heterogeneous acceleration in sectors like government, AI, healthcare, finance, and MPC.

openHiTLS

openHiTLS provides a lightweight, customizable cryptographic solution designed for diverse industries including cloud computing, big data, AI, and finance. The platform combines advanced algorithms with exceptional performance while maintaining robust security and reliability. Its flexible architecture ensures seamless compatibility across multiple programming languages. Through community collaboration and ecosystem development, openHiTLS accelerates the adoption of cryptographic security standards across various industries, while fostering a security-focused open source ecosystem centered around openEuler in order to provide users with a safer and more reliable digital environment.



Feature Description

Support for mainstream cryptographic protocols and algorithms

Mainstream international and Chinese cryptographic algorithms and protocols are supported. You can select appropriate cryptographic algorithms and protocols based on scenario requirements.

- Chinese cryptographic algorithms: SM2, SM3, and SM4
- International mainstream algorithms: AES, RSA, (EC)DSA, (EC)DH, SHA-3 and HMAC
- GB/T 38636-2020 TLCP, that is, the dual-certificate protocol
- TLS 1.2, TLS 1.3, and DTLS 1.2

Open architecture for cryptography applications in all scenarios

By leveraging an open architecture and implementing technological innovations in applications across the entire industry chain, a one-stop, full-scenario solution is provided for diverse industries.

- **Flexible southbound and northbound interfaces**: Unified northbound interfaces enable quick access for industry applications. Southbound devices widely deployed in various service systems are abstracted to streamline device utilization.
- **Multi-language compatibility**: The foreign function interfaces (FFIs) ensure multi-language compatibility, enabling one cipher suite to be used across various programming languages.

- **Wide applicability**: Cryptographic technology applied across various scenarios in the entire industry chain ensures high performance, security, and reliability of openHiTLS in different scenarios.

Hierarchical decoupling and on-demand tailoring to create a lightweight cipher suite

Hierarchical decoupling enables cryptographic algorithm software to achieve ultimate cost efficiency.

- **Hierarchical decoupling**: TLS, certificates, and algorithm functions are decoupled for on-demand combination, with layered optimization for algorithm abstraction, scheduling, and algorithm primitives.
- **Advanced abstract interfaces**: These interfaces prevent external interface changes caused by algorithm tailoring while reducing software footprint.
- **Ultimate cost efficiency**: Automatic management of feature dependencies and on-demand tailoring to trim down to the minimal implementation of PBKDF2 + AES (20 KB binary file size, 1 KB of heap memory, and 256 bytes of stack memory) are supported.

Agile architecture for cryptographic algorithms, addressing post-quantum migration

An innovative agile architecture for cryptographic algorithms enables rapid application migration and fast evolution of advanced algorithms.

- **Unified northbound interfaces**: Standardized and extensible interfaces are available for algorithms, meaning that interface changes due to algorithm switching can be avoided, which in turn means that extensive adaptation of new interfaces for upper-layer applications is not required.
- **Plugin-based management of algorithms**: The algorithm provider layer supports dynamic algorithm runtime loading.
- **Configurable algorithms**: Algorithm information can be obtained from configuration files, avoiding hard coding of algorithm identifiers.

Application Scenarios

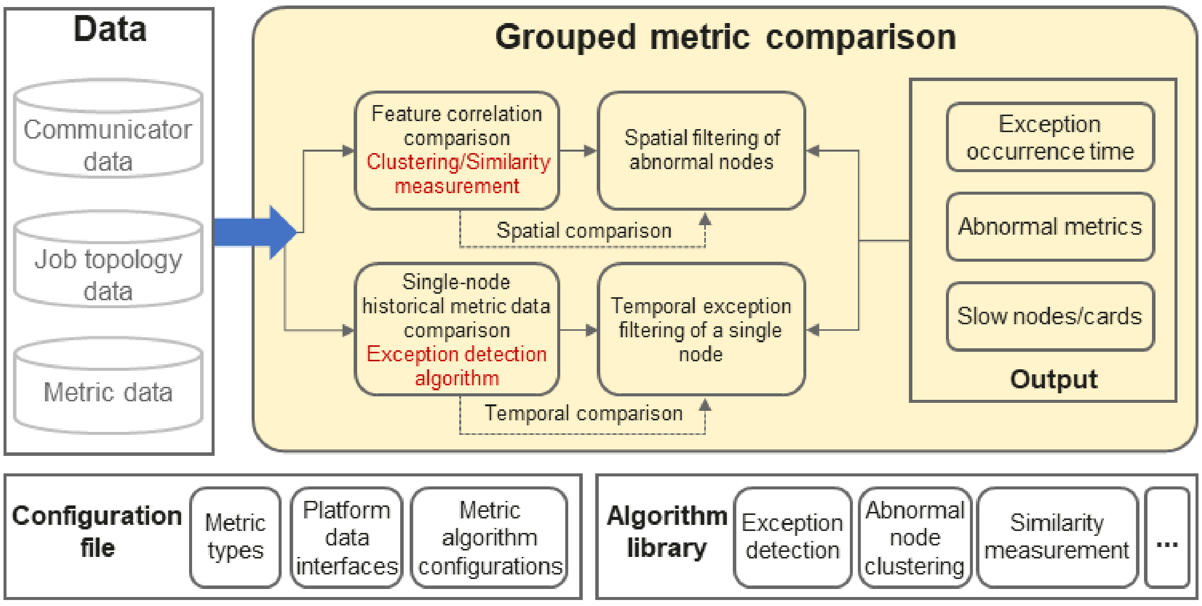
The openHiTLS cipher suite is applicable in a wide range of scenarios. Typical scenarios include:

- **Cloud computing**: With a modular and customizable design, openHiTLS can flexibly adapt to the security needs of cloud computing environments. It provides strong data encryption, identity authentication, and integrity check for cloud platforms, ensuring the security and integrity of data on the cloud.
- **Big data**: Data security and privacy protection are critical in big data processing. openHiTLS offers a variety of cryptographic protocols and algorithms for encrypted storage, transmission, and processing of big data, effectively preventing data leakage and unauthorized access.
- **Internet of Things (IoT)**: IoT devices are often constrained by limited resources. openHiTLS uses a lightweight and customizable software architecture to adapt to the low power consumption and resource constraints of IoT devices, ensuring secure and reliable communication and data protection.
- **Finance**: The financial industry has extremely high security requirements. openHiTLS supports Chinese cryptographic algorithms and complies with international security standards, offering strong data encryption, identity authentication, and payment security for financial systems to ensure the security and compliance of financial transactions.

Fail-slow Detection for Rapid Identification of Slow Nodes in AI Cluster Training

During the training process of AI clusters, performance degradation is inevitable, with numerous and complex causes. Existing solutions rely on log analysis after performance degradation occurs. However, it can take 3 to 4 days from log collection to root cause diagnosis and issue resolution on the live network. To address these pain points, an online slow node detection solution is offered. This solution allows for real-time monitoring of key system metrics and uses model- and data-driven algorithms to analyze the observed data and pinpoint the location of slow or degraded nodes. This facilitates system self-healing and fault rectification by O&M personnel.

Feature Description


Data
Communicator data
Job topology data
Metric data
Grouped metric comparison
Feature correlation comparison
Clustering/Similarity measurement
Spatial filtering of abnormal nodes
Spatial comparison
Single-node historical metric data comparison
Exception detection algorithm
Temporal exception filtering of a single node
Temporal comparison
Exception occurrence time
Abnormal metrics
Slow nodes/cards
Output
Configuration file
Metric types
Platform data interfaces
Metric algorithm configurations
Algorithm library
Exception detection
Abnormal node clustering
Similarity measurement
...

Grouped metric comparison helps detect slow nodes/cards in AI cluster training scenarios. This technology is implemented through gala-anteater and includes new components such as a configuration file, an algorithm library, and slow node comparison based on both time and space. The output includes the exception occurrence time, abnormal metrics, and IP addresses of slow nodes/cards. The technology improves system stability and reliability. The following features are provided:

- **Configuration file**: Contains the types of metrics to be observed, configuration parameters for the metric algorithms, and data interfaces, which are used to initialize the slow node detection algorithms.
- **Algorithm library**: Includes common time series exception detection algorithms, such as Streaming Peaks-over-Threshold (SPOT), k-sigma, abnormal node clustering, and similarity measurement.
- **Data**: Includes metric data, job topology data, and communicator data. Metric data indicates the time series of metrics, job topology data indicates the node information used in training jobs, and communicator data indicates the node connection relationships (including data parallelism, tensor parallelism, and pipeline parallelism).
- **Grouped metric comparison**: Supports spatial filtering of abnormal nodes and temporal exception filtering of a single node. Spatial filtering identifies abnormal nodes based on the

exception clustering algorithm, while temporal exception filtering determines whether a node is abnormal based on the historical data of the node.

Application Scenarios

gala-anteater supports slow node reporting, alarm display, and exception information display.

AI model training scenarios: This feature quickly detects slow nodes for training jobs in large-scale AI clusters, facilitating system self-healing and fault rectification by O&M personnel.

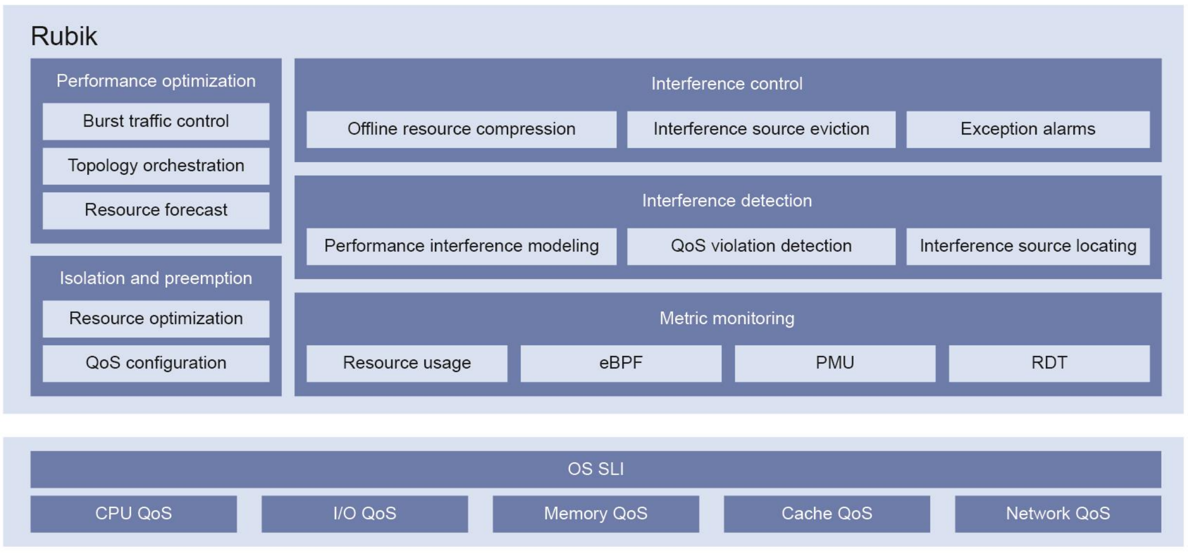
AI model inference scenarios: This feature helps detect performance degradation in multiple instances of a single model. By comparing application resources of multiple instances, it can quickly identify instances with performance degradation, facilitating inference job scheduling and improving resource utilization.

Rubik with Enhanced Collaboration for Hybrid Deployments

Cloud data centers face a widespread challenge of low resource utilization, typically below 20%. Improving this utilization has become a critical technical priority. Hybrid deployment—deploying workloads with different priorities together—offers an effective solution for increasing resource utilization. While hybrid deployment substantially improves cluster efficiency, it introduces resource contention that can impact the quality of service (QoS) of critical workloads. The key challenge lies in maintaining workload QoS while achieving higher resource utilization.

Rubik, openEuler's container hybrid deployment engine, implements an adaptive system for single-node computing optimization and QoS assurance. The engine maximizes node resource utilization without compromising the performance of critical workloads.

Feature Description



- **Cache and memory bandwidth control**: Limits the LLC and memory bandwidth of low-priority VMs. Currently, only static allocation is supported.
- **CPU interference control**: Supports CPU time slice preemption in microseconds, simultaneous multithreading (SMT) interference isolation, and anti-priority-inversion.
- **Memory resource preemption**: Terminates offline services first during node out-of memory (OOM) events to protect online service quality.

- **Memcg asynchronous memory reclamation**: Limits the total memory used by hybrid offline applications, and dynamically compresses the memory used by offline services when the online memory utilization increases.
- **quotaBurst traffic control**: When the CPU traffic of key online services is limited, the limit can be exceeded in a short period of time, ensuring the quality of online services.
- **Enhanced observation of PSI**: Collects pressure information at the cgroup v1 level, identifies and quantifies service interruption risks caused by resource contention, and improves hardware resource utilization.
- **iocost service I/O weight control**: Manages drive I/O rates for offline services to prevent bandwidth contention with online services.
- **Cycles per instruction (CPI) monitoring**: Tracks node pressure via CPI metrics to guide offline service eviction decisions.

New features

- **Node CPU/memory management**: Monitors resource levels and evicts offline services when necessary to maintain node stability.

Application Scenarios

To improve resource utilization, services are classified into high- and low-priority services based on latency sensitivity, and deployed accordingly. Latency-sensitive services are recommended for high-priority VMs, such as web services, high-performance databases, real-time rendering, and machine learning inference; while services not limited by latency can be used for low-priority VMs, such as video encoding, big data processing, offline rendering, and machine learning training.

Enhanced CFGO

The continuous growth in code volume has made front-end bound execution a common issue in processors, which impacts program performance. Feedback-directed optimization techniques in compilers can effectively solve this issue.

Continuous Feature Guided Optimization (CFGO) in GCC for openEuler and BiSheng Compiler refers to continuous feedback-directed optimization for multimodal files (source code and binaries) and the full lifecycle (compilation, linking, post-linking, runtime, OS, and libraries). The following techniques are included:

- **Code layout optimization**: Techniques such as basic block reordering, function rearrangement, and hot/cold separation are used to optimize the binary layout of the target program, improving I-cache and I-TLB hit rates.
- **Advanced compiler optimization**: Techniques such as inlining, loop unrolling, vectorization, and indirect calls enable the compiler to make more accurate optimization decisions.

Feature Description

CFGO comprises CFGO-PGO, CFGO-CSPGO, and CFGO-BOLT. Enabling these sub-features in sequence helps mitigate front-end bound execution and improve program runtime performance. To further enhance the optimization, you are advised to add the **-flto=auto** compilation option during CFGO-PGO and CFGO-CSPGO processes.

- CFGO-PGO

  Unlike conventional profile-guided optimization (PGO), CFGO-PGO uses AI for Compiler (AI4C) to enhance certain optimizations, including inlining, constant propagation, and devirtualization, to further improve performance.

- CFGO-CSPGO

  The profile in conventional PGO is context-insensitive, which may result in suboptimal optimization. By adding an additional CFGO-CSPGO instrumentation phase after PGO, runtime information from the inlined program is collected. This provides more accurate execution data for compiler optimizations such as code layout and register optimizations, leading to enhanced performance.
- CFGO-BOLT

  CFGO-BOLT adds optimizations such as software instrumentation for the AArch64 architecture and inlining optimization on top of the baseline version, driving further performance gains.

Application Scenarios

CFGO has good versatility and is suitable for C/C++ applications, such as databases and distributed storage, where the overall system performance bottleneck lies in the CPU front end, particularly in the front-end. It can typically achieve a performance improvement of 5% to 10%.

AI4C

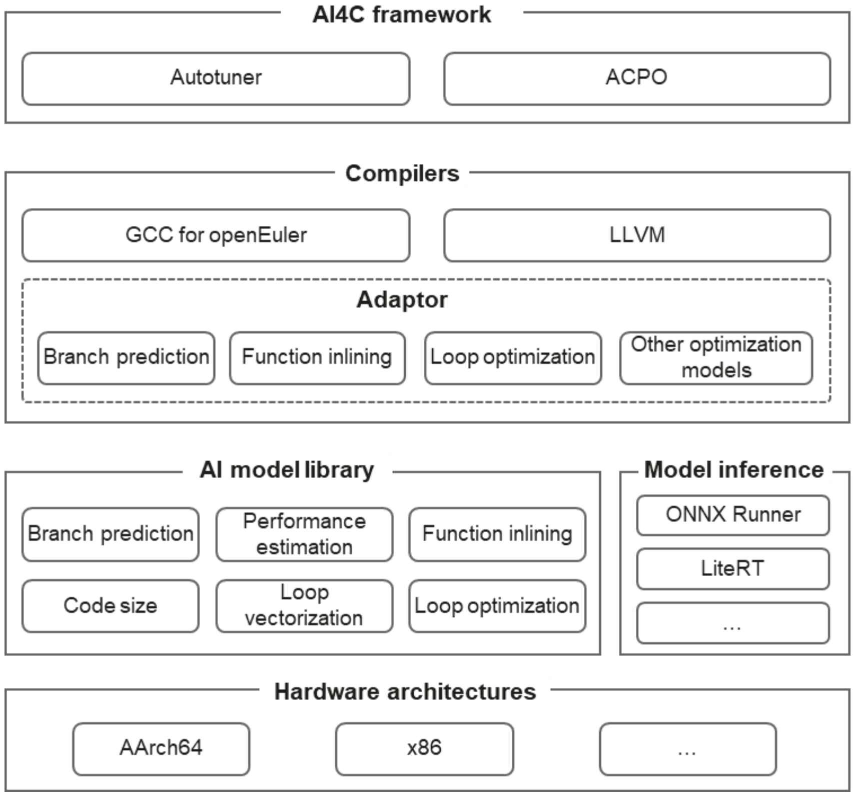
AI4C is an AI-assisted compiler optimization suite. It is a software framework that leverages AI technologies to optimize compiler options and key decisions during optimization passes. It aims to address two major challenges in compilers:

- **Difficult performance improvement**: Traditional compiler optimizations have long development cycles, and new optimization techniques are often incompatible with existing optimization processes, making it challenging to achieve the expected performance gains.
- **Low tuning efficiency**: When changes in hardware architecture or software application scenarios occur, significant human effort is required to adjust the cost model for compiler optimizations based on the new workloads, resulting in prolonged tuning times.

Feature Description

The AI4C framework provides two main modules: Autotuner for automatic tuning of compiler options and AI-enabled compiler-driven program optimization (ACPO).

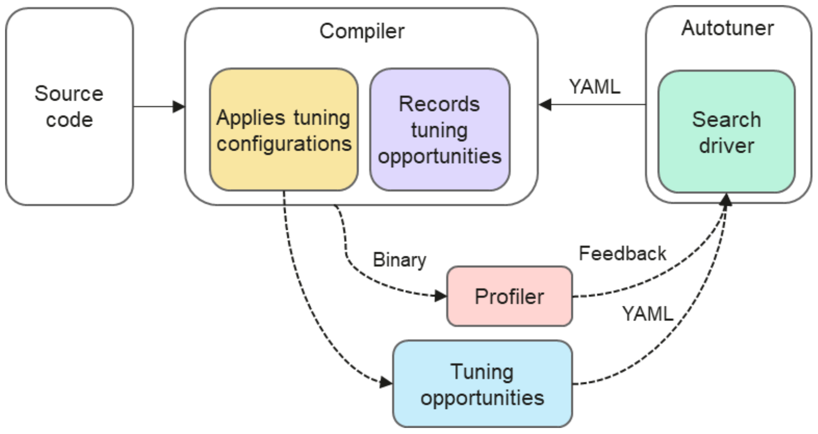
The software follows a three-layer architecture, as shown in the following figure. The AI4C framework at the upper layer drives the optimization process of a compiler at the middle layer. The Adaptor module of the compiler invokes the AI model library and model inference engine at the lower layer, using optimization feature data and hardware architecture parameters as inputs to run model inference. This results in the optimal setting for key parameters during compilation, thereby achieving compiler optimization.



- Autotuner

  The Autotuner of AI4C is developed based on OpenTuner (Ansel et al. 2015). It uses a plugin to drive the compiler to collect tuning parameters, adjusts key decision-making parameters (such as loop unrolling factors) using search algorithms, injects modifications into the compilation process via the plugin, and runs the compiled binary to gather feedback factors for iterative automatic tuning.

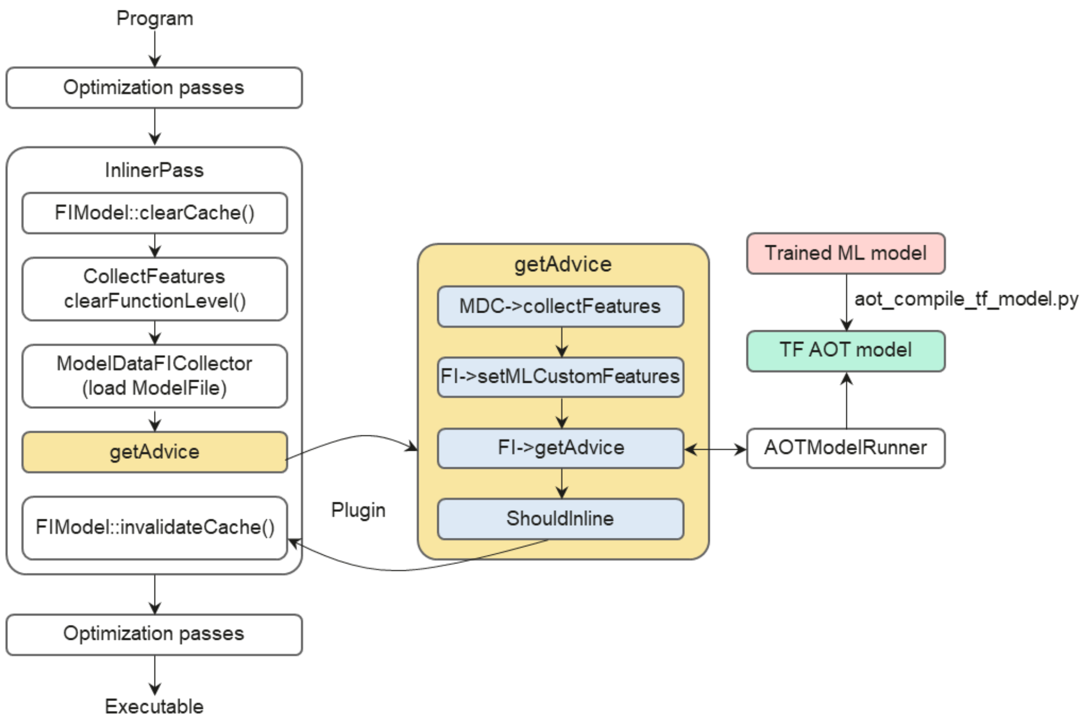
  - It integrates a series of search algorithms, with dynamic algorithm selection and shared search progress.
  - It supports user-configured YAML for custom search spaces and the extension of underlying search algorithms.
  - It enables fine-grained code block tuning and coarse-grained automatic tuning of compiler options.
  - It achieved performance gains ranging from 3% to 5% on benchmarks such as Cormark, Dhrystone, and Cbench.



- ACPO

  ACPO provides a comprehensive set of tools, libraries, and algorithms, which replace or enhance heuristic tuning decision-making algorithms in compilers and provide easy-to-use interfaces for compiler engineers to use AI models. During compiler optimization, plugins are used to extract structured input data from optimization passes as model input features. The **getAdvice** function runs the pre-trained model to obtain decision coefficients, and the compiler uses the model's decision results to replace specific heuristic decisions, thereby achieving better performance.

  - It decouples compilers from AI models and inference engines, helping algorithm developers focus on AI model development while reducing the costs of model application. It is compatible with multiple compilers, models, AI inference frameworks, and other mainstream products, offering hot update capabilities for AI models.
  - The implementation of optimization phases and processes, such as function inlining based on interprocedural analysis (IPA) and loop unrolling for register transfer language (RTL) generation, has resulted in significant gains.



Application Scenarios

AI4C provides optimization models and tuning plugins to achieve compiler optimizations for different applications. Developers can develop optimization models based on the performance bottlenecks of various applications and inject these models into the compilation process through AI4C to boost performance. Currently, AI4C is mainly used in backend fields such as databases in the Internet industry.

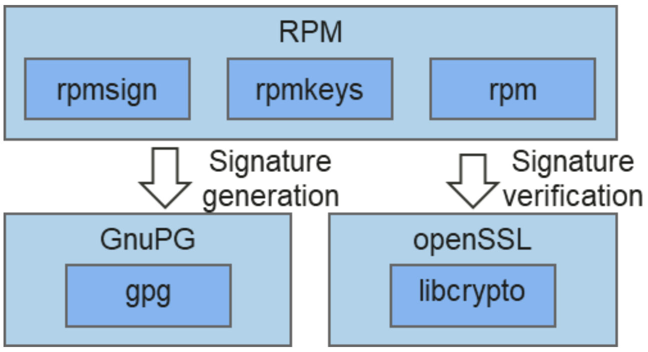
- RocksDB
- MySQL
- Doris
- Nginx
- Python
- ...

SM Digital Signatures for RPMs

According to relevant security standards in China, certain application scenarios require the use of Chinese cryptographic algorithms to ensure the authenticity and integrity of critical executable program sources. openEuler currently uses RPM for software package management, with package signatures based on the openPGP standard. openEuler 24.03 LTS SP1 incorporates the SM2 signing algorithm and SM3 digest algorithm into the RPM mechanism.

Feature Description

Based on the RPM component and the GnuPG2 signing tool the component invokes, this feature enables Chinese cryptographic algorithms within the existing openPGP signature system. The following software packages are required:



In signature generation, you can run the **gpg** command to generate an SM2 signing private key and a certificate, and then run the **rpmsign** command to add a digital signature based on the SM2 and SM3 algorithms to a specified RPM package.

In signature verification, you can run the **rpm** command to import the verification certificate and verify the digital signature of the RPM package to validate its authenticity and integrity.

Application Scenarios

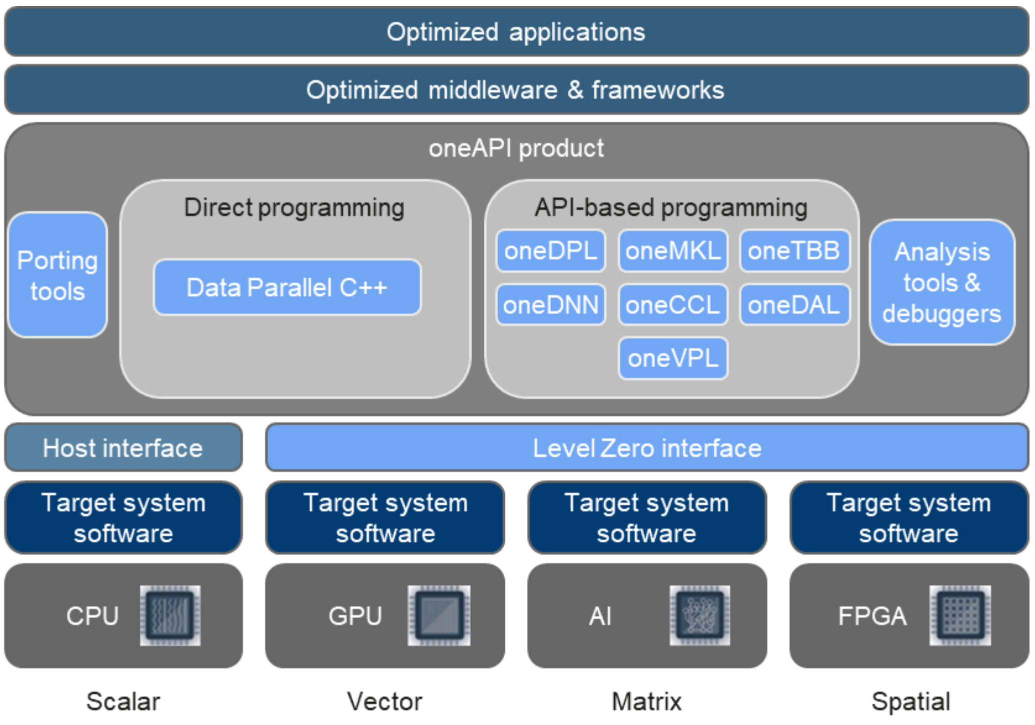
You can build software package verification capabilities based on Chinese cryptographic algorithms to meet cryptographic security requirements in certain compliance scenarios in China.

oneAPI

The Unified Acceleration (UXL) Foundation is promoting an open, unified standard accelerator software ecosystem. oneAPI, as the initial project, aims to provide a cross-industry, open, and unified standard programming model to deliver consistent development experience for heterogeneous accelerators, including those for CPUs, GPUs, FPGAs, and specialized accelerators. The oneAPI standard extends existing developer programming models to empower cross-architecture programming using a parallel programming language (Data Parallel C++), a group of accelerator software libraries, and a foundational hardware abstraction interface (Intel® oneAPI Level Zero). This approach supports a wide range of accelerator hardware and processor platforms. To ensure compatibility and improve development efficiency, the oneAPI standard provides various open, cross-platform, and easy-to-use developer software suites.

Feature Description

To fully support oneAPI, the Intel® oneAPI Base Toolkit (Base Kit) and runtime container images are integrated into openEuler 24.03 LTS. Since openEuler 24.03 LTS SP1, openEuler natively supports the adaptation and integration of oneAPI foundational libraries, including the dependencies required for oneAPI runtimes, Intel's graphics acceleration compiler, OpenCL™ Runtimes, and oneAPI Level Zero API that is compatible with platforms like x86_64 and AArch64. To comprehensively support Data Parallel C++ and API-based programming on accelerator libraries, various oneAPI software packages have been adapted and validated on openEuler. You can add the official DNF/Yum repositories of oneAPI to openEuler to install and update all required runtime dependencies, developer tools, and debugging utilities.



Application Scenarios

- **Cross-platform programming**: The unified programming framework of oneAPI allows programs written in Data Parallel C++ to be quickly re-compiled for different platforms. In addition, oneAPI, as a cross-architecture programming model that delivers both high performance and usability, can combine the capabilities of heterogeneous accelerators like CPUs, GPUs, NPUs, and FPGAs. This makes oneAPI suitable for a broad array of cross-platform high-performance computing (HPC) scenarios.

- **AI and deep learning**: oneAPI has been integrated with mainstream AI frameworks, allowing for seamless adaptation to heterogeneous accelerators through the oneAPI Level Zero API. This enables AI frameworks and models to fully utilize the acceleration capabilities of the hardware.
- **HPC, scientific computing, and simulation**: oneAPI offers a set of mathematical libraries and parallel computing tools designed to accelerate numerical computations such as linear algebra and differential equations. Moreover, oneAPI can be used to create simulation models for complex systems and accelerate computations using hardware (such as the Advanced Matrix Extensions, or AMX, for Intel Xeon processors) in scientific research in mathematics, physics, and engineering, and in industrial fields. oneAPI is compatible with industry-standard acceleration libraries, making it ideal for accelerating industrial software like CAE, thereby improving the speed and accuracy of simulation analysis.

Repository

https://gitee.com/openeuler/intel-oneapi

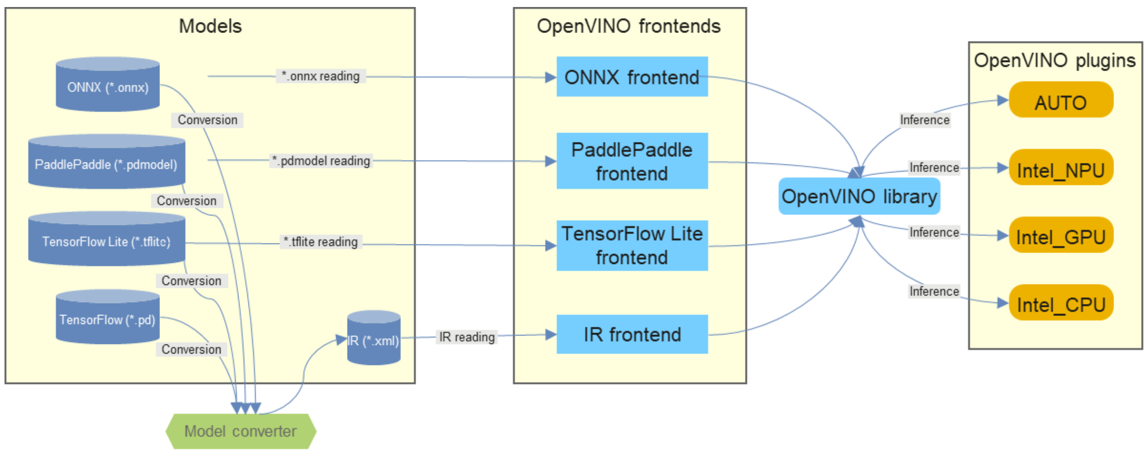
OpenVINO

OpenVINO is an open source AI toolkit and runtime library that enhances deep learning models from major frameworks. It streamlines the deployment and inference of AI workloads across processors and accelerators on Intel and other platforms including Arm. Beginning with openEuler 24.03 LTS SP1, native OpenVINO integration is provided, granting full access to OpenVINO computing capabilities on openEuler.

Feature Description

OpenVINO converts and optimizes models trained in popular frameworks to run efficiently on diverse hardware in local, edge, or cloud environments. Popular frameworks include TensorFlow, PyTorch, ONNX, and PaddlePaddle.

- **Pre-trained model library, model conversion, and optimization**: Open Model Zoo provides hundreds of open source pre-trained models ready for immediate use with OpenVINO. OpenVINO converts models from various deep learning frameworks into intermediate representations (IRs) for optimization. Using the Neural Network Compression Framework (NNCF), OpenVINO performs model pruning and sparsity compression to reduce size and boost inference speed, while its built-in performance analysis tools identify bottlenecks and optimize model performance.
- **Hardware-based acceleration**: Optimized models can use the parallel computing features from Intel and other platforms as the plugin architecture integrates third-party hardware acceleration solutions.
- **Runtime**: The inference engine of OpenVINO connects hardware platforms and OSs with computing languages (C, C++, Python) to enable asynchronous inference and boost throughput.
- **GenAI Support**: The OpenVINO GenAI library streamlines model inference by abstracting away complex generation processes and reducing the amount of code. This extension to the OpenVINO runtime provides specialized pipelines and methods for generative AI models, enabling quick deployment of new capabilities.



Application Scenarios

OpenVINO accelerates inference and streamlines deployment for mainstream trained models across local, edge, and cloud environments.

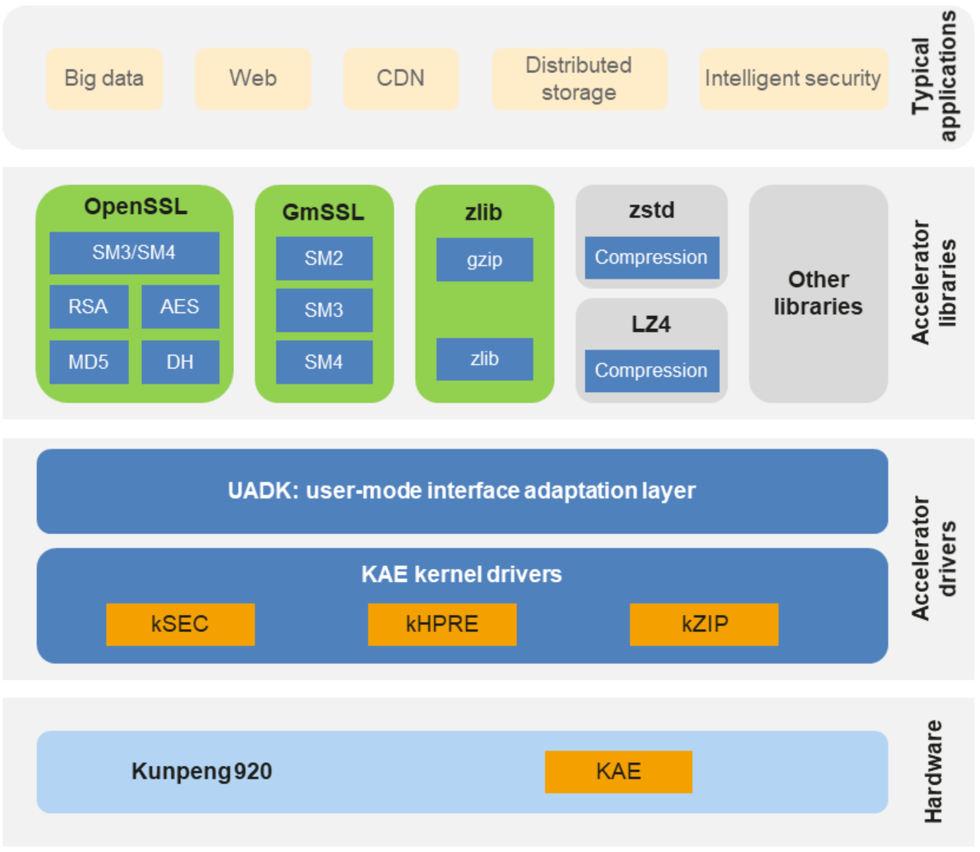
- **AI PC framework**: The OpenVINO open source framework is designed for inference optimization and model deployment. Positioned as platform agnostic, the framework supports multiple OSs, hardware platforms, and programming languages, enabling deep integration of various pre-trained models and providing core AI capabilities from the system level to the application layer for AI PCs.
- **Generative AI**: The GenAI API makes it quick and easy to deploy AI applications by integrating existing models for text-to-image generation, speech recognition, transcription, and vision-language processing.
- **Industry and edge AI**: By providing specialist development kits for edge AI, the framework streamlines development of applications in retail, manufacturing, healthcare, and other sectors.

Repository

https://gitee.com/openeuler/intel-openvino

KAE

KAE is an acceleration solution based on the Kunpeng 920 processor. It contains the KAE encryption and decryption module and the KAEzip compression and decompression module, which accelerate SSL and TLS applications and data compression, reduce processor usage, and boost processor efficiency. In addition, the application layer of KAE masks the internal implementation details, thereby allowing users to quickly migrate services through the standard interfaces of OpenSSL and zlib. The following figure shows the logical architecture of KAE:



The following table describes KAE modules:

Module	Description
UADK	User-mode interface adaptation layer of KAE, providing user-mode APIs
ksec/khpre/kzip	Kernel-mode device drivers of KAE
openSSL	OpenSSL library adapted to KAE to accelerate encryption and decryption
zlib	zlib library adapted to KAE to accelerate compression, providing standard open source zlib APIs

Feature Description

The KAE encryption and decryption module implements the RSA, SM3, SM4, DH, MD5, and AES algorithms. It provides high-performance symmetric and asymmetric encryption and decryption based on the lossless user-mode driver framework. It is compatible with OpenSSL 1.1.1*x* and supports both synchronous and asynchronous mechanisms. KAE supports the following algorithms:

- Digest algorithms SM3 and MD5, supporting asynchronous models.

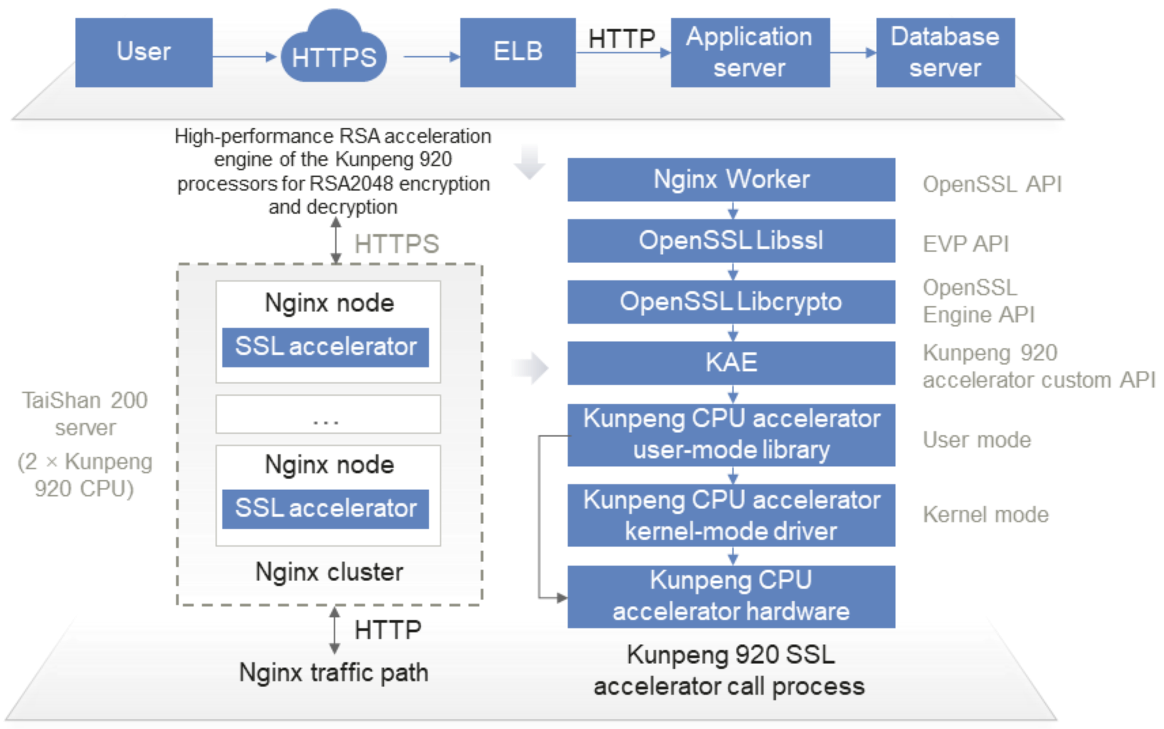
- Symmetric encryption algorithm SM4, supporting asynchronous models and CTR, XTS, CBC, ECB, and OFB modes.
- Symmetric encryption algorithm AES, supporting asynchronous models and ECB, CTR, XTS, and CBC modes.
- Asymmetric algorithm RSA, supporting asynchronous models and key sizes 1024, 2048, 3072, and 4096.
- Key negotiation algorithm DH, supporting asynchronous models and key sizes 768, 1024, 1536, 2048, 3072, and 4096.

KAEzip is the compression and decompression module of KAE. It implements the Deflate algorithm and works with the lossless user-mode driver framework to provide high-performance gzip or zlib APIs. KAE can be used to improve application performance in different scenarios. For example, in distributed storage scenarios, the zlib library accelerates data compression and decompression.

- zlib and gzip formats are supported, complying with RFC1950 and RFC1952 specifications.
- The synchronous mode is supported.
- A single Kunpeng 920 processor can use KAEzip to achieve a maximum compression bandwidth of 7 GB/s and a maximum decompression bandwidth of 8 GB/s.
- The supported compression ratio is around 2, which is the same as zlib 1.2.11.

Application Scenarios

KAE is well-suited to scenarios such as SSL acceleration for web applications and compressed storage for HDFS and distributed storage solutions. As shown in the following figure, in web application acceleration scenarios, KAE offers an OpenSSL compatibility engine that delivers hardware acceleration for encryption algorithms like RSA, AES, SM3, and SM4. Applications directly access HTTPS acceleration capabilities of KAE through standard OpenSSL APIs, requiring no code modifications.


User
HTTPS
ELB
HTTP
Application server
Database server
High-performance RSA acceleration engine of the Kunpeng 920 processors for RSA2048 encryption and decryption
HTTPS
Nginx node
SSL accelerator
...
Nginx node
SSL accelerator
Nginx cluster
TaiShan 200 server
(2 × Kunpeng 920 CPU)
HTTP
Nginx traffic path
Nginx Worker
OpenSSL Libssl
OpenSSL Libcrypto
KAE
Kunpeng CPU accelerator user-mode library
Kunpeng CPU accelerator kernel-mode driver
Kunpeng CPU accelerator hardware
Kunpeng 920 SSL accelerator call process
OpenSSL API
EVP API
OpenSSL Engine API
Kunpeng 920 accelerator custom API
User mode
Kernel mode

8 Copyright Statement

All materials or contents contained in this document are protected by the copyright law, and all copyrights are owned by openEuler, except for the content cited by other parties. Without a prior written permission of the openEuler community or other parties concerned, no person or organization shall reproduce, distribute, reprint, or publicize any content of this document in any form; link to or transmit the content through hyperlinks; upload the content to other servers using the "method of images"; store the content in information retrieval systems; or use the content for any other commercial purposes. For non-commercial and personal use, the content of the website may be downloaded or printed on condition that the content is not modified and all rights statements are reserved.

9 Trademarks

All trademarks and logos used and displayed on this document are all owned by the openEuler community, except for trademarks, logos, and trade names that are owned by other parties. Without the written permission of the openEuler community or other parties, any content in this document shall not be deemed as granting the permission or right to use any of the aforementioned trademarks and logos by implication, no objection, or other means. Without prior written consent, no one is allowed to use the name, trademark, or logo of the openEuler community in any form.

10 Appendixes

Appendix 1: Setting Up the Development Environment

Environment Setup	URL
Downloading and installing openEuler	https://openeuler.org/en/download/
Preparing the development environment	https://gitee.com/openeuler/community/blob/master/en/contributors/prepare-environment.md
Building a software package	https://gitee.com/openeuler/community/blob/master/en/contributors/package-install.md

Appendix 2: Security Handling Process and Disclosure

Security Issue Disclosure	URL
Security handling process	https://gitee.com/openeuler/security-committee/blob/master/security-process-en.md
Security disclosure	https://gitee.com/openeuler/security-committee/blob/master/security-disclosure-en.md